\*&---------------------------------------------------------------------\*  
\*& Report ZCL\_GLOBAL\_CONSTRUCTOR\_CALLING  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZCL\_GLOBAL\_CONSTRUCTOR\_CALLING.  
START-OF-SELECTION.  
  
  
data: globalobj type REF TO ZCL\_VAISH\_GLOBAL\_CONSTRUCTOR.  
create object  globalobj.

\*&---------------------------------------------------------------------\*  
\*& Report ZCL\_VAISH\_GLMETHODS\_CLASS\_PROG  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zcl\_vaish\_glmethods\_class\_prog.  
DATA obj-2 TYPE REF TO zcl\_vaish\_gmethods\_cls.  
CREATE OBJECT obj-2.  
  
START-OF-SELECTION.  
\*call method obj-2->objectmethod. "old method  
  obj-2->objectmethod( ). "new method  
  SKIP 2.  
\*call method ZCL\_VAISH\_GMETHODS\_CLS=>classmethod.  
  zcl\_vaish\_GMETHODS\_CLS=>classmethod( ).  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*local method\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
CLASS zcl\_local DEFINITION.  
  PUBLIC SECTION.  
    METHODS first .  
    CLASS-METHODS second.  
  
  
ENDCLASS.  
  
START-OF-SELECTION.  
  
CLASS zcl\_local IMPLEMENTATION.  
  METHOD first.  
    WRITE:'first name'.  
  ENDMETHOD.  
  METHOD second.  
    WRITE 'second method'.  
  ENDMETHOD.  
ENDCLASS.  
  
START-OF-SELECTION.  
  DATA obj-4 TYPE REF TO zcl\_local.  
  CREATE OBJECT obj-4.  
  
  obj-4->first( ).  
  zcl\_local=>second( ).

\*&---------------------------------------------------------------------\*  
\*& Report ZCL\_VAISH\_GL\_CLASS\_PROG  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZCL\_VAISH\_GL\_CLASS\_PROG.  
\*DATA: obj-1 type REF TO ZCL\_VAISH\_CLS\_PRGM. "declaring object as class  
\*create OBJECT obj-1.  
\*WRITE : / obj-1->I\_A,   "calling instance through object  
\*Zcl\_vaish\_cls\_prgm=>S\_B. "calling static attribute through class.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*local declaration\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
CLASS zcl\_lA DEFINITION.  
  PUBLIC SECTION.  
  DATA: I\_A TYPE I value '100'.  
  class-DATA S\_A type I value '200'.  
  
  
  
  ENDCLASS.  
  DATA :obj-1 type REF TO zcl\_lA.  
  create object Obj-1.  
  write:/ obj-1->I\_A,  
 zcl\_lA=>S\_A.

\*&---------------------------------------------------------------------\*  
\*& Report ZCL\_VAISH\_GL\_FIANL\_PRGM  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZCL\_VAISH\_GL\_FIANL\_PRGM.  
  
data: lt\_tab type zvaish\_tt\_emp.  
  
data lv\_empid type zvaish\_emp\_table-emp\_id.  
select-OPTIONS s\_empid for lv\_empid.  
data obj type REF TO ZCL\_VAISH\_FINAL\_GBL\_CLSMETHODS.  
create object obj.  
CALL METHOD obj->empmethod  
  EXPORTING  
    empidlow                 = '101'  
    empidhigh                = '105'  
IMPORTING  
  output                   = lt\_tab  
\*  EXCEPTIONS  
\*    out\_of\_boundaryexception = 1  
\*    others                   = 2  
        .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
CL\_DEMO\_OUTPUT=>DISPLAY( LT\_TAB ).

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ADOBESALES\_PROG  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ADOBESALES\_PROG.  
  
  
DATA : gv\_vbeln type vbak-vbeln,  
      gv\_func\_name type funcname,  
      gs\_customer type ZVAISH\_CUSTOMER\_STR,  
      gs\_output\_param type sfpoutputparams,  
      gt\_final type zvaish\_tt\_vbak2.  
  
SELECT-OPTIONS s\_vbeln FOR gv\_vbeln OBLIGATORY.  
  
AT SELECTION-SCREEN.  
  SELECT SINGLE VBELN  
    FROM VBAK  
    INTO GV\_VBELN  
    WHERE VBELN IN S\_VBELN.  
  
    IF SY-SUBRC IS NOT INITIAL.  
      MESSAGE e017(pord\_messages). "sap defined messages.  
      ENDIF.  
  
      START-OF-SELECTION.  
      SELECT VBELN, VBAK~erdat,vbtyp, vbak~kunnr, land1,adrnr  
        from vbak  
        INNER JOIN KNA1 ON KNA1~KUNNR = VBAK~KUNNR  
        INTO TABLE @DATA(gt\_vbak)  
        where vbeln in @s\_vbeln.  
  
        IF sy-subrc is INITIAL.  
          select vbeln,posnr,matnr,kwmeng,vrkme  
            from vbap  
            into table @DATA(gt\_vbap)  
            FOR ALL ENTRIES IN @gt\_vbak  
            where vbeln = @gt\_vbak-vbeln.  
  
            if sy-subrc IS NOT INITIAL.  
              CLEAR: gt\_vbak, gt\_vbap.  
              MESSAGE e017(prod\_messages).  
              ENDIF.  
  
  
              IF gt\_vbak is not initial.  
  
                SORT gt\_vbak by vbeln.  
                sort gt\_vbap by vbeln.  
                DATA(gv\_index) = VALUE syst\_tabix( ).  
                LOOP AT gt\_vbak ASSIGNING FIELD-SYMBOL(<lfs\_vbak>).  
                  APPEND INITIAL LINE TO gt\_final ASSIGNING FIELD-SYMBOL(<gfs\_final>). "appending empty line  
                  MOVE-CORRESPONDING <lfs\_vbak> TO <gfs\_final>.  
  
                  READ TABLE gt\_vbap TRANSPORTING NO FIELDS  
                  WITH KEY vbeln = <lfs\_vbak>-vbeln BINARY SEARCH.  
                  if sy-subrc is INITIAL.  
                    gv\_index = sy-tabix.  
                    LOOP AT gt\_vbap ASSIGNING FIELD-SYMBOL(<lfs\_vbap>) FROM gv\_index.  
                       IF <lfs\_vbap>-vbeln NE <lfs\_vbak>-vbeln.  
                         EXIT.  
                         ENDIF.  
                         APPEND INITIAL LINE TO <gfs\_final>-item ASSIGNING FIELD-SYMBOL(<lfs\_items>).  
                         MOVE-CORRESPONDING <lfs\_vbap> TO <lfs\_items>.  
                         ENDLOOP.  
                         ENDIF.  
                         ENDLOOP.  
                         endif.  
  
  
CALL FUNCTION 'FP\_JOB\_OPEN'  
  CHANGING  
    ie\_outputparams       =   gs\_output\_param  
\* EXCEPTIONS  
\*   CANCEL                = 1  
\*   USAGE\_ERROR           = 2  
\*   SYSTEM\_ERROR          = 3  
\*   INTERNAL\_ERROR        = 4  
\*   OTHERS                = 5  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
                         TRY.  
                           CALL FUNCTION 'FP\_FUNCTION\_MODULE\_NAME'  
                             EXPORTING  
                               i\_name                     = 'ZVAISH\_ADOBEFORM\_SALES'  
                         IMPORTING  
                              E\_FUNCNAME                 = gv\_func\_name  
\*                              E\_INTERFACE\_TYPE           =  
\*                              EV\_FUNCNAME\_INBOUND        =  
                                     .  
  
  
                           CATCH cx\_root.  
                             ENDTRY.  
  
  
  
    IF gv\_func\_name IS NOT INITIAL.  
      gs\_output\_param-device = 'LOCAL'  . "  'LOCL'.  
      gs\_output\_param-nodialog = abap\_true.  
      gs\_output\_param-preview = abap\_true.  
  
  
      CALL FUNCTION '/1BCDWB/SM00000633'  
        EXPORTING  
\*         /1BCDWB/DOCPARAMS        =  
          it\_tab                   = GT\_FINAL  
\*       IMPORTING  
\*         /1BCDWB/FORMOUTPUT       =  
\*       EXCEPTIONS  
\*         USAGE\_ERROR              = 1  
\*         SYSTEM\_ERROR             = 2  
\*         INTERNAL\_ERROR           = 3  
\*         OTHERS                   = 4  
                .  
      IF sy-subrc <> 0.  
\* Implement suitable error handling here  
      ENDIF.  
  
  
  
CALL FUNCTION 'FP\_JOB\_CLOSE'  
\* IMPORTING  
\*   E\_RESULT             =  
\* EXCEPTIONS  
\*   USAGE\_ERROR          = 1  
\*   SYSTEM\_ERROR         = 2  
\*   INTERNAL\_ERROR       = 3  
\*   OTHERS               = 4  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
endif.  
  
endif.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ADOBESALES\_PROG1  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ADOBESALES\_PROG1.  
  
  
DATA : gv\_vbeln type vbak-vbeln,  
      gv\_func\_name type funcname,  
      gs\_customer type ZVAISH\_CUSTOMER\_STR,  
      gs\_output\_param type sfpoutputparams,  
      gt\_final type zvaish\_tt\_vbak2.  
  
SELECT-OPTIONS s\_vbeln FOR gv\_vbeln OBLIGATORY.  
  
AT SELECTION-SCREEN.  
  SELECT SINGLE VBELN  
    FROM VBAK  
    INTO GV\_VBELN  
    WHERE VBELN IN S\_VBELN.  
  
    IF SY-SUBRC IS NOT INITIAL.  
      MESSAGE e017(pord\_messages). "sap defined messages.  
      ENDIF.  
  
      START-OF-SELECTION.  
      SELECT VBELN, VBAK~erdat,vbtyp, vbak~kunnr, land1,adrnr  
        from vbak  
        INNER JOIN KNA1 ON KNA1~KUNNR = VBAK~KUNNR  
        INTO TABLE @DATA(gt\_vbak)  
        where vbeln in @s\_vbeln.  
  
        IF sy-subrc is INITIAL.  
          select vbeln,posnr,matnr,kwmeng,vrkme  
            from vbap  
            into table @DATA(gt\_vbap)  
            FOR ALL ENTRIES IN @gt\_vbak  
            where vbeln = @gt\_vbak-vbeln.  
  
            if sy-subrc IS NOT INITIAL.  
              CLEAR: gt\_vbak, gt\_vbap.  
              MESSAGE e017(prod\_messages).  
              ENDIF.  
  
  
              IF gt\_vbak is not initial.  
  
                SORT gt\_vbak by vbeln.  
                sort gt\_vbap by vbeln.  
                DATA(gv\_index) = VALUE syst\_tabix( ).  
                LOOP AT gt\_vbak ASSIGNING FIELD-SYMBOL(<lfs\_vbak>).  
                  APPEND INITIAL LINE TO gt\_final ASSIGNING FIELD-SYMBOL(<gfs\_final>). "appending empty line  
                  MOVE-CORRESPONDING <lfs\_vbak> TO <gfs\_final>.  
  
                  READ TABLE gt\_vbap TRANSPORTING NO FIELDS  
                  WITH KEY vbeln = <lfs\_vbak>-vbeln BINARY SEARCH.  
                  if sy-subrc is INITIAL.  
                    gv\_index = sy-tabix.  
                    LOOP AT gt\_vbap ASSIGNING FIELD-SYMBOL(<lfs\_vbap>) FROM gv\_index.  
                       IF <lfs\_vbap>-vbeln NE <lfs\_vbak>-vbeln.  
                         EXIT.  
                         ENDIF.  
                         APPEND INITIAL LINE TO <gfs\_final>-item ASSIGNING FIELD-SYMBOL(<lfs\_items>).  
                         MOVE-CORRESPONDING <lfs\_vbap> TO <lfs\_items>.  
                         ENDLOOP.  
                         ENDIF.  
                         ENDLOOP.  
                         endif.  
  
  
CALL FUNCTION 'FP\_JOB\_OPEN'  
  CHANGING  
    ie\_outputparams       =   gs\_output\_param  
\* EXCEPTIONS  
\*   CANCEL                = 1  
\*   USAGE\_ERROR           = 2  
\*   SYSTEM\_ERROR          = 3  
\*   INTERNAL\_ERROR        = 4  
\*   OTHERS                = 5  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
                         TRY.  
                           CALL FUNCTION 'FP\_FUNCTION\_MODULE\_NAME'  
                             EXPORTING  
                               i\_name                     = 'ZVAISH\_ADOBEFORM\_SALESLOGO1'  
                         IMPORTING  
                              E\_FUNCNAME                 = gv\_func\_name  
\*                              E\_INTERFACE\_TYPE           =  
\*                              EV\_FUNCNAME\_INBOUND        =  
                                     .  
  
  
                           CATCH cx\_root.  
                             ENDTRY.  
CALL FUNCTION '/1BCDWB/SM00000792'  
  EXPORTING  
\*   /1BCDWB/DOCPARAMS        =  
    it\_tab                   = gt\_final  
  
\* IMPORTING  
\*   /1BCDWB/FORMOUTPUT       =  
\* EXCEPTIONS  
\*   USAGE\_ERROR              = 1  
\*   SYSTEM\_ERROR             = 2  
\*   INTERNAL\_ERROR           = 3  
\*   OTHERS                   = 4  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
  
  
  
CALL FUNCTION 'FP\_JOB\_CLOSE'  
\* IMPORTING  
\*   E\_RESULT             =  
\* EXCEPTIONS  
\*   USAGE\_ERROR          = 1  
\*   SYSTEM\_ERROR         = 2  
\*   INTERNAL\_ERROR       = 3  
\*   OTHERS               = 4  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
  
endif.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ADOBE\_FORM1\_PROGM  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zvaish\_adobe\_form1\_progm.  
  
INCLUDE zvaish\_top.  
  
PARAMETERS : P\_ebeln TYPE ebeln.  
  
START-OF-SELECTION.  
  
  wa\_outputparams-device = 'PRINTER'.  
  wa\_outputparams-dest = 'LP01'.  
  wa\_outputparams-nodialog = 'X'.  
  wa\_outputparams-preview = 'X'.  
  
  CALL FUNCTION 'FP\_JOB\_OPEN'  
    CHANGING  
      ie\_outputparams = wa\_outputparams  
\*     EXCEPTIONS  
\*     CANCEL          = 1  
\*     USAGE\_ERROR     = 2  
\*     SYSTEM\_ERROR    = 3  
\*     INTERNAL\_ERROR  = 4  
\*     OTHERS          = 5  
    .  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
  
  CALL FUNCTION 'FP\_FUNCTION\_MODULE\_NAME'  
    EXPORTING  
      i\_name     = 'ZVAISH\_ADOBE\_FORM1'  
    IMPORTING  
      e\_funcname = gv\_fm\_name  
\*     E\_INTERFACE\_TYPE           =  
\*     EV\_FUNCNAME\_INBOUND        =  
    .  
  
  wa\_docparams-langu = 'E'.  
  wa\_docparams-country = 'IN'.  
  
  
  SELECT SINGLE ebeln ebelp matnr bukrs werks  
    FROM ekpo  
    INTO ( gv\_ebeln, gv\_ebelp, gv\_matnr, gv\_bukrs, gv\_werks )  
  WHERE ebeln EQ p\_ebeln.  
  
  
    CALL FUNCTION       GV\_FM\_NAME            "'/1BCDWB/SM00000117'  
      EXPORTING  
      /1BCDWB/DOCPARAMS        = wa\_docparams  
        ebeln                    = gv\_ebeln  
        ebelp                    = gv\_ebelp  
        matnr                    = gv\_matnr  
        bukrs                    = gv\_bukrs  
        werks                    = gv\_werks  
\*     IMPORTING  
\*       /1BCDWB/FORMOUTPUT       =  
\*     EXCEPTIONS  
\*       USAGE\_ERROR              = 1  
\*       SYSTEM\_ERROR             = 2  
\*       INTERNAL\_ERROR           = 3  
\*       OTHERS                   = 4  
              .  
    IF sy-subrc <> 0.  
\* Implement suitable error handling here  
    ENDIF.  
  
  
  
  
  
  CALL FUNCTION 'FP\_JOB\_CLOSE'  
\*       IMPORTING  
\*         E\_RESULT             =  
\*       EXCEPTIONS  
\*         USAGE\_ERROR          = 1  
\*         SYSTEM\_ERROR         = 2  
\*         INTERNAL\_ERROR       = 3  
\*         OTHERS               = 4  
    .  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ADOBE\_VBAK\_PRGM  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zvaish\_adobe\_vbak\_prgm.  
INCLUDE zvaish\_vbak\_top.  
  
SELECT-OPTIONS s\_vbeln FOR lv\_vbeln.  
  
START-OF-SELECTION.  
  
  wa\_outputparams-device = 'PRINTER'.  
  wa\_outputparams-dest = 'LP01'.  
  wa\_outputparams-nodialog ='X'.  
  wa\_outputparams-preview = 'X'.  
  
  
  CALL FUNCTION 'FP\_JOB\_OPEN'  
    CHANGING  
      ie\_outputparams = wa\_outputparams  
\* EXCEPTIONS  
\*     CANCEL          = 1  
\*     USAGE\_ERROR     = 2  
\*     SYSTEM\_ERROR    = 3  
\*     INTERNAL\_ERROR  = 4  
\*     OTHERS          = 5  
    .  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
  
  CALL FUNCTION 'FP\_FUNCTION\_MODULE\_NAME'  
    EXPORTING  
      i\_name     = 'ZVAISH\_ADOBE\_TABLE\_FORM12'  
    IMPORTING  
      e\_funcname = gv\_fm\_name  
\*     E\_INTERFACE\_TYPE           =  
\*     EV\_FUNCNAME\_INBOUND        =  
    .  
  
  wa\_docparams-langu = 'E'.  
  wa\_docparams-country = 'IN'.  
  
  SELECT vbeln ernam erzet erdat  
   INTO TABLE LT\_TAB FROM vbak WHERE vbeln IN s\_vbeln.  
  
CALL FUNCTION '/1BCDWB/SM00000295'  
  EXPORTING  
  /1BCDWB/DOCPARAMS        = wa\_docparams  
    gt\_tab                   = LT\_TAB  
\* IMPORTING  
\*   /1BCDWB/FORMOUTPUT       =  
\* EXCEPTIONS  
\*   USAGE\_ERROR              = 1  
\*   SYSTEM\_ERROR             = 2  
\*   INTERNAL\_ERROR           = 3  
\*   OTHERS                   = 4  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
  
  CALL FUNCTION 'FP\_JOB\_CLOSE'  
\*           IMPORTING  
\*             E\_RESULT             =  
\*           EXCEPTIONS  
\*             USAGE\_ERROR          = 1  
\*             SYSTEM\_ERROR         = 2  
\*             INTERNAL\_ERROR       = 3  
\*             OTHERS               = 4  
    .  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.

REPORT ZVAISH\_ALV\_DISP\_LOGO.  
INCLUDE ZVAISH\_ALV\_MAT\_TOP.  
  
  
select-OPTIONS s\_matnr for lv\_matnr.  
  
INITIALIZATION.  
s\_matnr-low = '1'.  
s\_matnr-high = '100'.  
append s\_matnr.  
clear: lv\_matnr, wa\_fieldcat, lt\_fieldcat.  
  
at SELECTION-SCREEN.  
   select single matnr  
     into lv\_matnr  
     from mara  
     where matnr in s\_matnr.  
  
     if sy-subrc eq 0.  
       write: 'records are there'.  
       else.  
         write: 'records are not there'.  
         ENDIF.  
  
  
START-OF-SELECTION.  
perform get\_mat\_data.  
perform get\_mat\_alv.  
  
end-of-selection.  
  
perform display\_mat\_alv.  
  
  
FORM get\_mat\_data .  
  
select matnr mbrsh mtart meins brgew ntgew  
  into table lt\_tab  
  from mara  
  where matnr in s\_matnr.  
  
  if sy-subrc eq 0.  
    write 'records are found'.  
    else.  
      write 'records are not found'.  
      ENDIF.  
  
ENDFORM.  
  
  
  
FORM get\_mat\_alv.  
  
 wa\_fieldcat-col\_pos = '1'.  
  wa\_fieldcat-seltext\_m = 'material number'.  
   wa\_fieldcat-fieldname = 'matnr'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
     append wa\_fieldcat to lt\_fieldcat.  
  
       wa\_fieldcat-col\_pos = '2'.  
  wa\_fieldcat-seltext\_m = 'industry'.  
   wa\_fieldcat-fieldname = 'mbrsh'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '3'.  
  wa\_fieldcat-seltext\_m = 'order number'.  
   wa\_fieldcat-fieldname = 'mtart'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '4'.  
  wa\_fieldcat-seltext\_m = 'base unit of measure'.  
   wa\_fieldcat-fieldname = 'meins'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '5'.  
  wa\_fieldcat-seltext\_m = 'gross weight'.  
   wa\_fieldcat-fieldname = 'brgew'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
  
wa\_fieldcat-col\_pos = '6'.  
wa\_fieldcat-fieldname = 'ntgew'.  
wa\_fieldcat-seltext\_l = 'net weight'.  
wa\_fieldcat-tabname = 'lt\_tab'.  
    append wa\_fieldcat to lt\_fieldcat.  
  
  
  
ENDFORM.  
  
FORM display\_mat\_alv.  
  
  TY\_layout-no\_colhead = 'X'.  
  TY\_LAYOUT-no\_vline = 'X'.  
   TY\_layout-no\_hline = 'X'.  
  
CALL FUNCTION 'REUSE\_ALV\_GRID\_DISPLAY'  
  
EXPORTING  
\*   I\_INTERFACE\_CHECK                 = ' '  
\*   I\_BYPASSING\_BUFFER                = ' '  
\*   I\_BUFFER\_ACTIVE                   = ' '  
  I\_CALLBACK\_PROGRAM                = SY-REPID  
\*   I\_CALLBACK\_PF\_STATUS\_SET          = ' '  
\*   I\_CALLBACK\_USER\_COMMAND           = ' '  
I\_CALLBACK\_TOP\_OF\_PAGE            = 'LOGO'  
\*   I\_CALLBACK\_HTML\_TOP\_OF\_PAGE       = ' '  
\*   I\_CALLBACK\_HTML\_END\_OF\_LIST       = ' '  
\*   I\_STRUCTURE\_NAME                  =  
\*   I\_BACKGROUND\_ID                   = ' '  
\*   I\_GRID\_TITLE                      =  
\*   I\_GRID\_SETTINGS                   =  
 IS\_LAYOUT                         =  TY\_layout  
   IT\_FIELDCAT                       = LT\_FIELDCAT  
\*   IT\_EXCLUDING                      =  
\*   IT\_SPECIAL\_GROUPS                 =  
\*   IT\_SORT                           =  
\*   IT\_FILTER                         =  
\*   IS\_SEL\_HIDE                       =  
\*   I\_DEFAULT                         = 'X'  
\*   I\_SAVE                            = ' '  
\*   IS\_VARIANT                        =  
\*   IT\_EVENTS                         =  
\*   IT\_EVENT\_EXIT                     =  
\*   IS\_PRINT                          =  
\*   IS\_REPREP\_ID                      =  
\*   I\_SCREEN\_START\_COLUMN             = 0  
\*   I\_SCREEN\_START\_LINE               = 0  
\*   I\_SCREEN\_END\_COLUMN               = 0  
\*   I\_SCREEN\_END\_LINE                 = 0  
\*   I\_HTML\_HEIGHT\_TOP                 = 0  
\*   I\_HTML\_HEIGHT\_END                 = 0  
\*   IT\_ALV\_GRAPHICS                   =  
\*   IT\_HYPERLINK                      =  
\*   IT\_ADD\_FIELDCAT                   =  
\*   IT\_EXCEPT\_QINFO                   =  
\*   IR\_SALV\_FULLSCREEN\_ADAPTER        =  
\*   O\_PREVIOUS\_SRAL\_HANDLER           =  
\*   O\_COMMON\_HUB                      =  
\* IMPORTING  
\*   E\_EXIT\_CAUSED\_BY\_CALLER           =  
\*   ES\_EXIT\_CAUSED\_BY\_USER            =  
  TABLES  
    t\_outtab                          = LT\_TAB  
\* EXCEPTIONS  
\*   PROGRAM\_ERROR                     = 1  
\*   OTHERS                            = 2  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
  
  ENDFORM.  
  
  FORM LOGO.  
    DATA: wa\_header type slis\_listheader,  
          lt\_header type slis\_T\_listheader.  
  
  
    wa\_header-typ = 'S'.  
    wa\_header-info = 'ust global'.  
    wa\_header-key = 'hyderabad'.  
    APPEND wa\_header to lt\_header.  
  
    wa\_header-typ = 'H'.  
    wa\_header-key = sy-datum.  
    wa\_header-info = sy-uzeit.  
      APPEND wa\_header to lt\_header.  
  
    CALL FUNCTION 'REUSE\_ALV\_COMMENTARY\_WRITE'  
      EXPORTING  
  
        it\_list\_commentary = lt\_header  
      i\_logo             = 'ZVAISH\_LOGO'  
\*       I\_END\_OF\_LIST\_GRID       =  
\*       I\_ALV\_FORM               =  
              .  
  
  
  
    ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ALV\_GRID\_DISPLAY  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ALV\_GRID\_DISPLAY.  
\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ALV\_LIST\_DISPLAY  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
  
INCLUDE ZVAISH\_ALV\_MAT\_TOP.  
  
SELECT-OPTIONS s\_matnr for lv\_matnr obligatory.  
  
INITIALIZATION.  
clear : wa\_fieldcat, lt\_fieldcat.  
s\_matnr-low = '1'.  
s\_matnr-high = '100'.  
APPEND s\_matnr.  
  
at SELECTION-SCREEN.  
  select single matnr  
    into lv\_matnr  
    from mara  
    where matnr in s\_matnr.  
  
    if sy-subrc eq 0.  
      write : 'records are found'.  
      else.  
        write : 'records are not found'.  
      ENDIF.  
  
  
  
START-OF-SELECTION.  
perform get\_mat\_data.  
perform get\_alv\_fcat.  
end-of-selection.  
perform dis\_mat\_alv.  
  
  
  
  
  
FORM get\_mat\_data .  
  
  select matnr mbrsh mtart meins brgew ntgew  
    into table lt\_tab  
    from mara  
    where matnr in s\_matnr.  
  
     if sy-subrc eq 0.  
       write: 'records are found'.  
       ELSe.  
         write:'records are not found'.  
       ENDIF.  
  
ENDFORM.  
  
  
  
  
  
FORM get\_alv\_fcat.  
  
  wa\_fieldcat-col\_pos = '1'.  
  wa\_fieldcat-seltext\_m = 'material number'.  
   wa\_fieldcat-fieldname = 'matnr'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
     append wa\_fieldcat to lt\_fieldcat.  
  
       wa\_fieldcat-col\_pos = '2'.  
  wa\_fieldcat-seltext\_m = 'industry'.  
   wa\_fieldcat-fieldname = 'mbrsh'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '3'.  
  wa\_fieldcat-seltext\_m = 'order number'.  
   wa\_fieldcat-fieldname = 'mtart'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '4'.  
  wa\_fieldcat-seltext\_m = 'base unit of measure'.  
   wa\_fieldcat-fieldname = 'meins'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '5'.  
  wa\_fieldcat-seltext\_m = 'gross weight'.  
   wa\_fieldcat-fieldname = 'brgew'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
  
wa\_fieldcat-col\_pos = '6'.  
wa\_fieldcat-fieldname = 'ntgew'.  
wa\_fieldcat-seltext\_l = 'net weight'.  
wa\_fieldcat-tabname = 'lt\_tab'.  
    append wa\_fieldcat to lt\_fieldcat.  
  
  
  
ENDFORM.  
  
  
  
  
FORM dis\_mat\_alv .  
  
  CALL FUNCTION 'REUSE\_ALV\_GRID\_DISPLAY'  
  EXPORTING  
\*     I\_BUFFER\_ACTIVE                = ' '  
 I\_CALLBACK\_PROGRAM             = ' ZVAISH\_ALV\_GRID\_DISPLAY'  
  
   IT\_FIELDCAT                    = lt\_fieldcat  
  
    TABLES  
      t\_outtab                       =  lt\_tab  
\*   EXCEPTIONS  
\*     PROGRAM\_ERROR                  = 1  
\*     OTHERS                         = 2  
            .  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ALV\_HIERARCHY  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ALV\_HIERARCHY.  
  
TYPES: BEGIN OF ty\_vbak,  
  FC(1), "to expand the output .it is not a header  
  vbeln TYPE vbeln\_va,  
  erdat TYPE erdat,  
  erzet TYPE erzet,  
  end of ty\_vbak.  
  
  DATA: lt\_vbak type TABLE of ty\_vbak,  
        wa\_vbak type ty\_vbak.  
  
TYPES: BEGIN OF ty\_vbap,  
  vbeln TYPE vbeln\_va,  
  posnr TYPE posnr\_va,  
  matnr TYPE matnr,  
  end of ty\_vbap.  
  
  DATA: lt\_vbap type table of ty\_vbap,  
        wa\_vbap type ty\_vbap,  
  
        lt\_fieldcat type slis\_t\_fieldcat\_alv,  
        wa\_fieldcat LIKE LINE OF  lt\_fieldcat,  
  
        ty\_layout type slis\_layout\_alv,  
        it\_key type slis\_keyinfo\_alv.  
  
TABLES VBAK.  
 select-OPTIONS s\_vbeln for vbak-VBELN.  
  
  
 START-OF-SELECTION.  
        perform get\_data.  
        perform get\_alv.  
 end-of-SELECTION.  
  
 perform hierarchydisplay\_alv.  
  
FORM get\_data.  
  
  select vbeln erdat erzet  
    from vbak  
    into CORRESPONDING FIELDS OF TABLE lt\_vbak  
    where vbeln in s\_vbeln.  
  
  select vbeln posnr matnr  
    from vbap  
    into table lt\_vbap  
    where vbeln in s\_vbeln.  
  
  
ENDFORM.  
  
FORM get\_alv .  
  wa\_fieldcat-col\_pos = '1'.  
  wa\_fieldcat-fieldname = 'VBELN'.  
  wa\_fieldcat-tabname = 'LT\_VBAK'.  
  wa\_fieldcat-ref\_tabname = 'VBAK'.  
  wa\_fieldcat-ref\_fieldname = 'VBELN'.  
  APPEND wa\_fieldcat TO lt\_fieldcat.  
  clear wa\_fieldcat.  
  
  wa\_fieldcat-col\_pos = '2'.  
  wa\_fieldcat-fieldname = 'ERDAT'.  
  wa\_fieldcat-tabname ='LT\_VBAK'.  
  wa\_fieldcat-ref\_fieldname = 'ERDAT'.  
  wa\_fieldcat-ref\_tabname = 'VBAK'.  
  append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '3'.  
  wa\_fieldcat-fieldname = 'ERZET'.  
  wa\_fieldcat-tabname = 'LT\_VBAK'.  
  wa\_fieldcat-ref\_fieldname = 'ERZET'.  
  wa\_fieldcat-ref\_tabname = 'VBAK'.  
  append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '1'.  
  wa\_fieldcat-fieldname = 'VBELN'.  
  wa\_fieldcat-tabname = 'LT\_VBAP'.  
  wa\_fieldcat-ref\_tabname = 'VBAP'.  
  wa\_fieldcat-ref\_fieldname = 'VBELN'.  
  APPEND wa\_fieldcat TO lt\_fieldcat.  
  clear wa\_fieldcat.  
  
     wa\_fieldcat-col\_pos = '2'.  
  wa\_fieldcat-fieldname = 'POSNR'.  
  wa\_fieldcat-tabname = 'LT\_VBAP'.  
  wa\_fieldcat-ref\_tabname = 'VBAP'.  
  wa\_fieldcat-ref\_fieldname = 'POSNR'.  
  APPEND wa\_fieldcat TO lt\_fieldcat.  
  clear wa\_fieldcat.  
  
     wa\_fieldcat-col\_pos = '3'.  
  wa\_fieldcat-fieldname = 'MATNR'.  
  wa\_fieldcat-tabname = 'LT\_VBAP'.  
  wa\_fieldcat-ref\_tabname = 'VBAP'.  
  wa\_fieldcat-ref\_fieldname = 'MATNR'.  
  APPEND wa\_fieldcat TO lt\_fieldcat.  
  clear wa\_fieldcat.  
  
  
  
ENDFORM.  
  
  
FORM hierarchydisplay\_alv .  
    it\_key-header01 = 'VBELN'.  
 ty\_layout-expand\_fieldname = 'FC'.  
  
  
  CALL FUNCTION 'REUSE\_ALV\_HIERSEQ\_LIST\_DISPLAY'  
    EXPORTING  
\*     I\_INTERFACE\_CHECK              = ' '  
    I\_CALLBACK\_PROGRAM             = SY-REPID  
\*     I\_CALLBACK\_PF\_STATUS\_SET       = ' '  
\*     I\_CALLBACK\_USER\_COMMAND        = ' '  
    IS\_LAYOUT                      = ty\_layout  
   IT\_FIELDCAT                    = lt\_fieldcat  
\*     IT\_EXCLUDING                   =  
\*     IT\_SPECIAL\_GROUPS              =  
\*     IT\_SORT                        =  
\*     IT\_FILTER                      =  
\*     IS\_SEL\_HIDE                    =  
\*     I\_SCREEN\_START\_COLUMN          = 0  
\*     I\_SCREEN\_START\_LINE            = 0  
\*     I\_SCREEN\_END\_COLUMN            = 0  
\*     I\_SCREEN\_END\_LINE              = 0  
\*     I\_DEFAULT                      = 'X'  
\*     I\_SAVE                         = ' '  
\*     IS\_VARIANT                     =  
\*     IT\_EVENTS                      =  
\*     IT\_EVENT\_EXIT                  =  
      i\_tabname\_header               = 'LT\_VBAK'  
      i\_tabname\_item                 = 'LT\_VBAP'  
\*     I\_STRUCTURE\_NAME\_HEADER        =  
\*     I\_STRUCTURE\_NAME\_ITEM          =  
      is\_keyinfo                     = it\_key  
  
\*     IS\_PRINT                       =  
\*     IS\_REPREP\_ID                   =  
\*     IS\_HIERSEQ\_SETTINGS            =  
\*     I\_BYPASSING\_BUFFER             =  
\*     I\_BUFFER\_ACTIVE                =  
\*     IR\_SALV\_HIERSEQ\_ADAPTER        =  
\*     IT\_EXCEPT\_QINFO                =  
\*     I\_SUPPRESS\_EMPTY\_DATA          = ABAP\_FALSE  
\*     I\_LIST\_DISPLAY\_ONLY            = ABAP\_FALSE  
\*     O\_COMMON\_HUB                   =  
\*   IMPORTING  
\*     E\_EXIT\_CAUSED\_BY\_CALLER        =  
\*     ES\_EXIT\_CAUSED\_BY\_USER         =  
    TABLES  
      t\_outtab\_header                = LT\_VBAK  
      t\_outtab\_item                  = LT\_vbap  
\*   EXCEPTIONS  
\*     PROGRAM\_ERROR                  = 1  
\*     OTHERS                         = 2  
            .  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ALV\_LIST\_DISPLAY  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ALV\_LIST\_DISPLAY.  
INCLUDE ZVAISH\_ALV\_MAT\_TOP.  
  
SELECT-OPTIONS s\_matnr for lv\_matnr obligatory.  
  
INITIALIZATION.  
clear : wa\_fieldcat, lt\_fieldcat.  
s\_matnr-low = '1'.  
s\_matnr-high = '100'.  
APPEND s\_matnr.  
  
at SELECTION-SCREEN.  
  select single matnr  
    into lv\_matnr  
    from mara  
    where matnr in s\_matnr.  
  
    if sy-subrc eq 0.  
      write : 'records are found'.  
      else.  
        write : 'records are not found'.  
      ENDIF.  
  
  
  
START-OF-SELECTION.  
perform get\_mat\_data.  
perform get\_alv\_fcat.  
end-of-selection.  
perform dis\_mat\_alv.  
  
  
  
  
  
FORM get\_mat\_data .  
  
  select matnr mbrsh mtart meins brgew ntgew  
    into table lt\_tab  
    from mara  
    where matnr in s\_matnr.  
  
     if sy-subrc eq 0.  
       write: 'records are found'.  
       ELSe.  
         write:'records are not found'.  
       ENDIF.  
  
ENDFORM.  
  
  
  
  
  
FORM get\_alv\_fcat.  
  
  wa\_fieldcat-col\_pos = '1'.  
  wa\_fieldcat-seltext\_m = 'material number'.  
   wa\_fieldcat-fieldname = 'matnr'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
     append wa\_fieldcat to lt\_fieldcat.  
  
       wa\_fieldcat-col\_pos = '2'.  
  wa\_fieldcat-seltext\_m = 'industry'.  
   wa\_fieldcat-fieldname = 'mbrsh'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '3'.  
  wa\_fieldcat-seltext\_m = 'order number'.  
   wa\_fieldcat-fieldname = 'mtart'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '4'.  
  wa\_fieldcat-seltext\_m = 'base unit of measure'.  
   wa\_fieldcat-fieldname = 'meins'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '5'.  
  wa\_fieldcat-seltext\_m = 'gross weight'.  
   wa\_fieldcat-fieldname = 'brgew'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
  
wa\_fieldcat-col\_pos = '6'.  
wa\_fieldcat-fieldname = 'ntgew'.  
wa\_fieldcat-seltext\_l = 'net weight'.  
wa\_fieldcat-tabname = 'lt\_tab'.  
    append wa\_fieldcat to lt\_fieldcat.  
  
  
  
ENDFORM.  
  
  
  
  
FORM dis\_mat\_alv .  
  
  CALL FUNCTION 'REUSE\_ALV\_LIST\_DISPLAY'  
  EXPORTING  
\*     I\_BUFFER\_ACTIVE                = ' '  
 I\_CALLBACK\_PROGRAM             = ' ZVAISH\_ALV\_LIST\_DISPLAY'  
  
   IT\_FIELDCAT                    = lt\_fieldcat  
  
    TABLES  
      t\_outtab                       =  lt\_tab  
\*   EXCEPTIONS  
\*     PROGRAM\_ERROR                  = 1  
\*     OTHERS                         = 2  
            .  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ALV\_MERGE  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ALV\_MERGE.  
  
  
  
  
INCLUDE ZVAISH\_ALV\_MERGE\_TOP.  
  
select-OPTIONS s\_matnr for lv\_matnr.  
  
INITIALIZATION.  
s\_matnr-low = '1'.  
s\_matnr-high = '100'.  
append s\_matnr.  
clear: lv\_matnr, wa\_fieldcat, lt\_fieldcat.  
  
at SELECTION-SCREEN on s\_matnr.  
   select single matnr  
     into lv\_matnr  
     from mara  
     where matnr in s\_matnr.  
  
     if sy-subrc eq 0.  
       write: 'records are there'.  
       else.  
         write: 'records are not there'.  
         ENDIF.  
  
  
START-OF-SELECTION.  
perform get\_mat\_data.  
perform get\_mat\_alv.  
  
end-of-selection.  
  
perform display\_mat\_alv.  
  
  
FORM get\_mat\_data .  
  
select \*  
  into table lt\_tab  
  from mara  
  where matnr in s\_matnr.  
  
  if sy-subrc eq 0.  
    write 'records are found'.  
    else.  
      write 'records are not found'.  
      ENDIF.  
  
ENDFORM.  
  
  
  
FORM get\_mat\_alv.  
  
 CALL FUNCTION 'REUSE\_ALV\_FIELDCATALOG\_MERGE'  
EXPORTING  
   I\_PROGRAM\_NAME               = SY-REPID  
 I\_INTERNAL\_TABNAME           = 'LT\_TAB'  
 I\_STRUCTURE\_NAME             = 'MARA'  
\*    I\_CLIENT\_NEVER\_DISPLAY       = 'X'  
\*    I\_INCLNAME                   =  
\*    I\_BYPASSING\_BUFFER           =  
\*    I\_BUFFER\_ACTIVE              =  
   CHANGING  
     ct\_fieldcat                  = LT\_FIELDCAT  
\*  EXCEPTIONS  
\*    INCONSISTENT\_INTERFACE       = 1  
\*    PROGRAM\_ERROR                = 2  
\*    OTHERS                       = 3  
           .  
 IF sy-subrc <> 0.  
\* Implement suitable error handling here  
 ENDIF.  
  
  
  
  
ENDFORM.  
  
FORM display\_mat\_alv.  
  
  TY\_layout-no\_colhead = 'X'.  
  TY\_LAYOUT-no\_vline = 'X'.  
   TY\_layout-no\_hline = 'X'.  
  
CALL FUNCTION 'REUSE\_ALV\_GRID\_DISPLAY'  
  
EXPORTING  
\*   I\_INTERFACE\_CHECK                 = ' '  
\*   I\_BYPASSING\_BUFFER                = ' '  
\*   I\_BUFFER\_ACTIVE                   = ' '  
  I\_CALLBACK\_PROGRAM                = SY-REPID  
\*   I\_CALLBACK\_PF\_STATUS\_SET          = ' '  
\*   I\_CALLBACK\_USER\_COMMAND           = ' '  
\*I\_CALLBACK\_TOP\_OF\_PAGE            = 'LOGO'  
\*   I\_CALLBACK\_HTML\_TOP\_OF\_PAGE       = ' '  
\*   I\_CALLBACK\_HTML\_END\_OF\_LIST       = ' '  
\*   I\_STRUCTURE\_NAME                  =  
\*   I\_BACKGROUND\_ID                   = ' '  
\*   I\_GRID\_TITLE                      =  
\*   I\_GRID\_SETTINGS                   =  
 IS\_LAYOUT                         =  TY\_layout  
   IT\_FIELDCAT                       = LT\_FIELDCAT  
\*   IT\_EXCLUDING                      =  
\*   IT\_SPECIAL\_GROUPS                 =  
\*   IT\_SORT                           =  
\*   IT\_FILTER                         =  
\*   IS\_SEL\_HIDE                       =  
\*   I\_DEFAULT                         = 'X'  
\*   I\_SAVE                            = ' '  
\*   IS\_VARIANT                        =  
\*   IT\_EVENTS                         =  
\*   IT\_EVENT\_EXIT                     =  
\*   IS\_PRINT                          =  
\*   IS\_REPREP\_ID                      =  
\*   I\_SCREEN\_START\_COLUMN             = 0  
\*   I\_SCREEN\_START\_LINE               = 0  
\*   I\_SCREEN\_END\_COLUMN               = 0  
\*   I\_SCREEN\_END\_LINE                 = 0  
\*   I\_HTML\_HEIGHT\_TOP                 = 0  
\*   I\_HTML\_HEIGHT\_END                 = 0  
\*   IT\_ALV\_GRAPHICS                   =  
\*   IT\_HYPERLINK                      =  
\*   IT\_ADD\_FIELDCAT                   =  
\*   IT\_EXCEPT\_QINFO                   =  
\*   IR\_SALV\_FULLSCREEN\_ADAPTER        =  
\*   O\_PREVIOUS\_SRAL\_HANDLER           =  
\*   O\_COMMON\_HUB                      =  
\* IMPORTING  
\*   E\_EXIT\_CAUSED\_BY\_CALLER           =  
\*   ES\_EXIT\_CAUSED\_BY\_USER            =  
  TABLES  
    t\_outtab                          = LT\_TAB  
\* EXCEPTIONS  
\*   PROGRAM\_ERROR                     = 1  
\*   OTHERS                            = 2  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
  
  ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ALV\_POPUP  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ALV\_POPUP.  
  
  
INCLUDE ZVAISH\_ALV\_MAT\_TOP.  
  
SELECT-OPTIONS s\_matnr for lv\_matnr obligatory.  
  
INITIALIZATION.  
clear : wa\_fieldcat, lt\_fieldcat.  
s\_matnr-low = '1'.  
s\_matnr-high = '100'.  
APPEND s\_matnr.  
  
at SELECTION-SCREEN.  
  select single matnr  
    into lv\_matnr  
    from mara  
    where matnr in s\_matnr.  
  
    if sy-subrc eq 0.  
      write : 'records are found'.  
      else.  
        write : 'records are not found'.  
      ENDIF.  
  
  
  
START-OF-SELECTION.  
perform get\_mat\_data.  
perform get\_alv\_fcat.  
end-of-selection.  
perform dis\_mat\_alv.  
  
  
  
  
  
FORM get\_mat\_data .  
  
  select matnr mbrsh mtart meins brgew ntgew  
    into table lt\_tab  
    from mara  
    where matnr in s\_matnr.  
  
     if sy-subrc eq 0.  
       write: 'records are found'.  
       ELSe.  
         write:'records are not found'.  
       ENDIF.  
  
ENDFORM.  
  
  
  
  
  
FORM get\_alv\_fcat.  
  
  wa\_fieldcat-col\_pos = '1'.  
  wa\_fieldcat-seltext\_m = 'material number'.  
   wa\_fieldcat-fieldname = 'matnr'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
    wa\_fieldcat-emphasize = 'C711'.  
     append wa\_fieldcat to lt\_fieldcat.  
  
       wa\_fieldcat-col\_pos = '2'.  
  wa\_fieldcat-seltext\_m = 'industry'.  
   wa\_fieldcat-fieldname = 'mbrsh'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '3'.  
  wa\_fieldcat-seltext\_m = 'order number'.  
   wa\_fieldcat-fieldname = 'mtart'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '4'.  
  wa\_fieldcat-seltext\_m = 'base unit of measure'.  
   wa\_fieldcat-fieldname = 'meins'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '5'.  
  wa\_fieldcat-seltext\_m = 'gross weight'.  
   wa\_fieldcat-fieldname = 'brgew'.  
     wa\_fieldcat-tabname = 'lt\_tab'.  
         append wa\_fieldcat to lt\_fieldcat.  
  
  
wa\_fieldcat-col\_pos = '6'.  
wa\_fieldcat-fieldname = 'ntgew'.  
wa\_fieldcat-seltext\_l = 'net weight'.  
wa\_fieldcat-tabname = 'lt\_tab'.  
    append wa\_fieldcat to lt\_fieldcat.  
  
  
  
ENDFORM.  
  
  
  
  
FORM dis\_mat\_alv .  
  
CALL FUNCTION 'REUSE\_ALV\_POPUP\_TO\_SELECT'  
  EXPORTING  
 I\_TITLE                       = 'MATERIAL MASTER INFORMATION'  
\*   I\_SELECTION                   = 'X'  
\*   I\_ALLOW\_NO\_SELECTION          =  
 I\_ZEBRA                       = ' '  
\*   I\_SCREEN\_START\_COLUMN         = 0  
\*   I\_SCREEN\_START\_LINE           = 0  
\*   I\_SCREEN\_END\_COLUMN           = 0  
\*   I\_SCREEN\_END\_LINE             = 0  
\*   I\_CHECKBOX\_FIELDNAME          =  
\*   I\_LINEMARK\_FIELDNAME          =  
\*   I\_SCROLL\_TO\_SEL\_LINE          = 'X'  
    i\_tabname                     = 'lt\_tab'  
   I\_STRUCTURE\_NAME              = 'ls\_tab'  
  IT\_FIELDCAT                   = lt\_fieldcat  
\*   IT\_EXCLUDING                  =  
\*   I\_CALLBACK\_PROGRAM            =  
\*   I\_CALLBACK\_USER\_COMMAND       =  
\*   IS\_PRIVATE                    =  
\* IMPORTING  
\*   ES\_SELFIELD                   =  
\*   E\_EXIT                        =  
  tables  
    t\_outtab                      = lt\_tab  
\* EXCEPTIONS  
\*   PROGRAM\_ERROR                 = 1  
\*   OTHERS                        = 2  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ASSESMENTS  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ASSESMENTS.  
\*TYPES: BEGIN OF LS\_TAB,  
\*  MAKTX TYPE  MAKTX,  
\*MAKTG TYPE  MAKTG,  
\*  MATNR TYPE MATNR,  
\*  MEINS TYPE MEINS,  
\*  MTART TYPE MTART,  
\*  
\*  END OF LS\_TAB.  
\*  
\*  DATA: LT\_TAB TYPE STANDARD TABLE OF LS\_TAB,  
\*        WA\_TAB TYPE LS\_TAB.  
\*  SELECT-OPTIONS S\_MATNR for WA\_TAB-MATNR.  
  
\*  START-OF-SELECTION.  
\*  PERFORM CREATE-MARAMAKT.  
\*  PERFORM DISPLAY-MARAMAKT.  
\*\*&---------------------------------------------------------------------\*  
\*\*& Form CREATE-MARAMAKT  
\*\*&---------------------------------------------------------------------\*  
\*\*& text  
\*\*&---------------------------------------------------------------------\*  
\*\*& -->  p1        text  
\*\*& <--  p2        text  
\*\*&---------------------------------------------------------------------\*  
\*FORM create-maramakt .  
\*  
  
\*ENDFORM.  
  
\*START-OF-SELECTION.  
\*  
\*  SELECT \*  
\*    INTO TABLE LT\_TAB  
\*    FROM mara  
\*    WHERE matnr IN s\_matnr.  
\*  
\*  IF sy-subrc EQ 0.  
\*    WRITE : 'sales order info ', sy-dbcnt.  
\*  ELSE.  
\*    WRITE : 'sales order records are not found ', sy-dbcnt.  
\*  ENDIF.  
\*  
\*END-OF-SELECTION.  
\*  LOOP AT lt\_tab INTO wa\_tab.  
\*    WRITE : / wa\_tab-maktx,  
\*    wa\_tab-maktg,  
\*    wa\_tab-matnr,  
\*    wa\_tab-meins,  
\*    wa\_tab-mtart.  
\*  ENDLOOP.  
  
  
  
  
TYPES : BEGIN OF ls\_tab,  
  emp\_id type zemp\_id,  
  emp\_name type zvemp\_name,  
  emp\_address type zemp\_address,  
  dpt\_name type zvdpt\_name,  
  end of ls\_tab.  
  
  data: lt\_tab type STANDARD TABLE OF ls\_tab,  
        wa\_tab type ls\_tab.  
  
  
START-OF-SELECTION.  
  select emp\_id emp\_name emp\_address dpt\_name  
    into table lt\_tab  
    from zvaish\_view1.  
  
    end-of-selection.  
  
    loop at lt\_tab into wa\_tab.  
      write:/ wa\_tab-emp\_id,  
      wa\_tab-emp\_name,  
      wa\_tab-emp\_address,  
      wa\_tab-dpt\_name.  
      ENDLOOP.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_CLASSICAL\_REPORT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_CLASSICAL\_REPORT NO STANDARD PAGE HEADING LINE-COUNT 20(3).  
  
load-of-program.  
  
include ZVAISH\_GET\_VBAK\_TOP.  
  
SELECT-OPTIONS S\_vbeln for wa\_tab-vbeln.  
  
initialization.  
clear: lt\_tab,wa\_tab.  
  
s\_vbeln-low = '1'.  
s\_vbeln-high = '100'.  
append s\_vbeln.  
  
at SELECTION-SCREEN.  
  select single vbeln  
    into lv\_vbeln  
    from vbak  
    where vbeln in s\_vbeln.  
  
    if sy-subrc eq  0.  
      MESSAGE s000(zvaish\_messages).  
      else.  
        MESSAGE e001(zvaish\_messages).  
      endif.  
  
START-OF-SELECTION.  
perform get\_vbak\_data.  
end-of-selection.  
perform disp\_vbak\_data.  
  
data lv\_count type i.  
lv\_count = sy-linct - sy-linno.  
skip lv\_count.  
  
top-of-page.  
uline.  
write :/  'Client',  
20 'Sales Document',  
30 'Record Created On',  
40 'Entry time',  
50 'Name of Person Responsible for Creating the Object',  
60  'Quotation/Inquiry is Valid From'.  
uline.  
  
  
end-of-page.  
write :/  
  'date:', sy-datum,  
50 'list-no:',sy-pagno,  
70 'time:', sy-uzeit.  
\*&---------------------------------------------------------------------\*  
\*& Form get\_vbak\_data  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM get\_vbak\_data .  
  
select VBELN ERDAT ERZET ERNAM ANGDT kunnr  
  into table lt\_tab  
  from vbak  
  where vbeln in s\_vbeln.  
ENDFORM.  
  
  
FORM disp\_vbak\_data .  
 loop at lt\_tab into wa\_tab.  
     write :/ wa\_tab-VBELN ,  
wa\_tab-ERDAT,  
wa\_tab-ERZET,  
wa\_tab-ERNAM,  
wa\_tab-ANGDT,  
wa\_tab-kunnr.  
   ENDLOOP.  
  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_CTRL\_STMTS  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_CTRL\_STMTS.  
\*&---------------------------------------------------------------------\*  
  
  
\*\*\*\*\*\*\*iF and else stmt\*\*\*\*\*\*\*\*\*\*\*\*  
data lv\_a(15) type c value 'vaishnavi'.  
write lv\_a.  
if lv\_a eq 'vais'.  
  write:/ 'yes'.  
  else.  
    write:/'no'.  
  ENDIF.  
  
\*\*\*\*\*\*do stmt\*\*\*\*\*\*\*\*  
\*  do .  
\*    write :/  '1'.  
\*    ENDDO.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*while loop\*\*\*\*\*\*\*\*  
  
\*data l type i value 1.  
\*while l < 10.  
\*  write / l.  
\*  l = l + 1.  
\*  endwhile.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*exit comes out of the do loop\*\*\*\*\*\*\*\*\*\*\*\*  
  
\*  
\*  do 10 TIMES.  
\*write sy-index.  
\*    if sy-index eq 5.  
\*      exit.  
\*      ENDIF.  
\*write 'inside loop'.  
\*    ENDDO.  
\*    write 'out of loop'.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*stop just stops the loop without procedding further\*\*\*\*\*\*\*\*\*\*\*  
\*    do 10 times.  
\*      write sy-index.  
\*      if sy-index eq 5.  
\*        stop.  
\*        ENDIF.  
\*      ENDDO.  
\*      write 'out of the loop'.  
  
  
\*\*\*\*\*\*\*\*\*\*\*continue keyword\*\*\*\*\*\*\*\*  
\*do 10 times.  
\*  write sy-index.  
\*  if sy-index eq 5.  
\*    write sy-index.  
\*    continue.  
\*    endif.  
\*  enddo.  
\*  write 'out of the loop'.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*check \*\*\*\*\*\*\*\*\*\*  
\*  
\*do 10 times.  
\*  
\*if sy-index EQ 5.  
\*   write sy-index.  
\*   exit.  
\*  endif.  
\*  
\* check sy-index eq 2.  
\*    write sy-index.  
\*  
\*  enddo.  
\*  write ' out of the loop'.  
\*  
  
  
  
  
  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*case\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*PARAMETERS P\_A type i default 5.  
\*  
\*case P\_A.  
\*  when 1.  
\*    write:'thats me'.  
\*  
\*    when 2.  
\*      write:'thats my frnd'.  
\*      when others.  
\*        write: 'others'.  
\*  
\*  
\*ENDCASE.  
\*write / 'out of loop'.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_DBVIEW  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_DBVIEW.  
include zvaish\_apreport.  
  
  
  
  select-options  s\_aufnr for wa\_tab-aufnr.  
  
  
START-OF-SELECTION.  
  
  select AUFNR  
AUART  
AUTYP  
WERKS  
POSNR  
QUNUM  
QUPOS  
    into table lt\_tab  
    from ZVAISH\_AUFPO  
    where aufnr in s\_aufnr.  
  
  
    if sy-subrc eq 0.  
      write : / 'records found' , sy-dbcnt.  
      else.  
        write :'records not found', sy-dbcnt.  
        ENDIF.  
        skip 3.  
write:/ 'id',  
 10 'name',  
30 'city',  
 40 'region'.  
  
uline.  
  
  
     End-of-SELECTION.  
 format color col\_positive.  
    loop at lt\_tab into wa\_tab.  
      write :/ wa\_tab-aufnr,  
  
wa\_tab-AUART,  
wa\_tab-AUTYP,  
wa\_tab-WERKS,  
wa\_tab-POSNR,  
wa\_tab-QUNUM,  
wa\_tab-QUPOS.  
  
  
      ENDLOOP.  
format color off.  
  
back.  
skip 2.  
write  100  'ust'.  
back.  
skip 3.  
write 100 'next'.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_FINAL  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_FINAL.  
\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_FITAB  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
\*REPORT ZVAISH\_FITAB.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*final itab code\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
\*types: begin of ls\_tab,  
\*  lv\_cdtid type zcdt\_id,  
\*  lv\_cdtname type zcdt\_name,  
\*  lv\_cdtcity type zcdt\_city,  
\*  end of ls\_tab.  
\*  
\*  data: lv\_cdtid type zcdt\_id.  
\*  SELECT-OPTIONS s\_cdtid for lv\_cdtid.  
\*  PARAMETERS p\_city type zcandidate-cdt\_city.  
\*  
\*  data:lt\_tab type STANDARD TABLE OF ls\_tab,  
\*        wa\_tab type ls\_tab.  
\*  
\*  select cdt\_id cdt\_name cdt\_city  
\*    into table lt\_tab  
\*    from zcandidate  
\*    where cdt\_id in s\_cdtid  
\*    and cdt\_city eq p\_city.  
\*  
\*    loop at lt\_tab into wa\_tab.  
\*      write:/ wa\_tab-lv\_cdtid,  
\*      wa\_tab-lv\_cdtname,  
\*      wa\_tab-lv\_cdtcity.  
\*      ENDLOOP.  
\*  
  
\*  
\*TYPES : BEGIN OF ls\_tab,   " LOCAL STRS  
\*          kunnr TYPE kunnr ,  "  Customer Number.  
\*          name1 TYPE name1\_gp ,  " name  
\*          ort01 TYPE ort01\_gp,  " city  
\*          land1 TYPE land1\_gp,  " Country/Region Key  
\*        END OF ls\_tab.  
\*  
\*DATA : LT\_tAB   TYPE STANDARD TABLE OF ls\_tab, " ITAB  
\*       wa\_tab   TYPE LS\_tAB,   " WORK AREA  
\*       lv\_kunnr TYPE kunnr.  
\*  
\*  
\*SELECT-OPTIONS s\_kunnr FOR lv\_kunnr OBLIGATORY.  " 1  to 100  
\*PARAMETERS p\_land1 TYPE kna1-land1.  " us  
\*  
\*START-OF-SELECTION.  
\*  
\*  SELECT  kunnr name1 ort01 land1  
\*    INTO TABLE LT\_tAB  
\*         FROM kna1  
\*          WHERE kunnr IN s\_kunnr AND  
\*                land1 EQ p\_land1.  
\*  
\*  IF sy-subrc EQ 0.  
\*    WRITE : / 'records are found in kna1 table ', sy-dbcnt.  
\*  ELSE.  
\*    WRITE : / 'records are not found in kna1 table ', sy-dbcnt.  
\*  ENDIF.  
\*\*\*\*\*SKIP .  
\*SKIP 3.  
\*  
\*WRITE : / 'ROW NO',  
\*         10 'CUST NO',  
\*         30 'CUST NAME',  
\*         50 'CITY',  
\*         65 'Region Key'.  
\*  
\*ULINE.  
\*  
\*end-of-SELECTION.  
\*FORMAT COLOR COL\_POSITIVE.  
\*  
\*  LOOP AT lt\_Tab INTO WA\_Tab.  " b to W  
\*    WRITE: / SY-TABIX,  
\*            10 WA\_tab-kunnr, " COLOR 1,   " W-f  
\*            30 WA\_tab-name1,  "  COLOR 2,  
\*            50 WA\_tab-ort01, "  COLOR COL\_GROUP,  
\*            65 WA\_tab-land1. "  COLOR COL\_POSITIVE.  
\*  ENDLOOP.  
\*  
\*FORMAT COLOR OFF.  
\*  
\*BACK.  
\*SKIP 2.  
\*    WRITE /120 'UST HYD'.  
\*BACK.  
\*SKIP 2.  
\*    WRITE /130 'UST USA'.  
  
  
  
types: begin of ls\_tab,  
  kunnr type kunnr,  
  name1 type name1\_gp,  
  ort01 type ort01\_gp,  
  land1 type land1\_gp,  
  end of ls\_tab.  
  
  
    data: lt\_tab type STANDARD TABLE OF ls\_tab,  
        wa\_tab type ls\_tab,  
        lv\_kunnr type kunnr.  
  
  SELECT-OPTIONS s\_kunnr for lv\_kunnr.  
  PARAMETERS p\_land1 type kna1-land1.  
  
START-OF-SELECTION.  
  
  select kunnr name1 land1 ort01  
    into table lt\_tab  
    from kna1  
    where kunnr in s\_kunnr  
    and land1 eq p\_land1.  
  
    if sy-subrc eq 0.  
      write : / 'records found' , sy-dbcnt.  
      else.  
        write :'records not found', sy-dbcnt.  
        ENDIF.  
        skip 3.  
write:/ 'id',  
 10 'name',  
30 'city',  
 40 'region'.  
  
uline.  
  
  
     End-of-SELECTION.  
 format color col\_positive.  
    loop at lt\_tab into wa\_tab.  
      write :/ wa\_tab-kunnr,  
      wa\_tab-name1,  
      wa\_tab-land1,  
      wa\_tab-ort01.  
  
      ENDLOOP.  
format color off.  
  
back.  
skip 2.  
write  100  'ust'.  
back.  
skip 3.  
write 100 'next'.  
  
  
  
  
  
  
\*

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_FORALL\_JOINS  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zvaish\_forall\_joins.  
  
INCLUDE zvaish\_forall\_joins\_top.  
  
DATA : lv\_kunnr TYPE kunnr.  
SELECT-OPTIONS s\_kunnr FOR lv\_kunnr.  
  
SELECT  
  kna1~kunnr  
  kna1~land1  
  kna1~name1  
  kna1~name2  
  
  knb1~bukrs  
  knb1~pernr  
  INTO TABLE lt\_tab  
  FROM kna1 INNER JOIN knb1 ON  kna1~kunnr = knb1~kunnr  
  WHERE kna1~kunnr IN s\_kunnr.  
  
IF NOT lt\_tab[] IS INITIAL. "if not 0 or empty  
  SELECT  vbeln  erdat  erzet  ernam kunnr  
    INTO TABLE lt\_tab2  
    FROM vbak  
    FOR ALL ENTRIES IN lt\_tab  
    WHERE kunnr eq lt\_tab-kunnr.  
  
  DELETE ADJACENT DUPLICATES FROM lt\_tab COMPARING kunnr.  
  SORT lt\_tab2 BY kunnr.  
  LOOP AT lt\_tab INTO wa\_tab.  
    READ TABLE lt\_tab2 INTO wa\_tab2 WITH KEY kunnr = wa\_tab-kunnr BINARY SEARCH.  
  
  
    IF sy-subrc = 0.  
      MOVE : wa\_tab-kunnr TO wa\_tabf-kunnr,  
      wa\_tab-kunnr TO wa\_tabf-kunnr,  
      wa\_tab-land1 TO wa\_tabf-land1,  
      wa\_tab-name1 TO wa\_tabf-name1,  
      wa\_tab-name2 TO wa\_tabf-name2,  
      wa\_tab-bukrs TO wa\_tabf-bukrs,  
      wa\_tab-pernr TO wa\_tabf-pernr,  
      wa\_tab2-vbeln TO wa\_tabf-vbeln,  
      wa\_tab2-erdat TO wa\_tabf-erdat,  
      wa\_tab2-erzet TO wa\_tabf-erzet,  
      wa\_tab2-ernam TO wa\_tabf-ernam.  
      APPEND wa\_tabf TO lt\_tabf.  
  
  ELSE.  
    CONTINUE.  
  ENDIF.  
    ENDLOOP.  
  
  LOOP AT lt\_tabf INTO wa\_tabf.  
  
    WRITE:/ wa\_tabf-kunnr,  
         wa\_tabf-land1,  
         wa\_tabf-name1,  
         wa\_tabf-name2,  
         wa\_tabf-bukrs,  
         wa\_tabf-pernr,  
         wa\_tabf-vbeln,  
         wa\_tabf-erdat,  
         wa\_tabf-erzet,  
         wa\_tabf-ernam.  
  
  ENDLOOP.  
  endif.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_GET\_CURSOR  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_GET\_CURSOR.  
 DATA: column1(10) type c value 'id',  
       column2(10) type c value 'name',  
       column3(10) type c value 'city',  
       click(10) type c.  
  
 START-OF-SELECTION.  
  
 FORMAT color 7.  
 write 'basic or first report'.  
  
 format HOTSPOT on.  
 write: / column1, column2,column3 . "hand symbol will be visible only on these two  
 format hotspot off.  
  
 AT LINE-SELECTION.  
  
get cursor field click.  
 if click = TEXT-001.  """"'COLUMN1'.  
   write : /  
    'user interacted with column 1'.  
   elseif click = 'COLUMN2'.  
     write :'user interacted with column2'.  
     elseif click = 'COLUMN3'.  
       write: 'user interacted with column3'.  
       endif.  
  
  
  
\*\*\*\*\*# 1  
\*DATA : var1(10) TYPE C VALUE 'Field1',  
\*      var2(10) TYPE C VALUE 'Field2',  
\*      var3(10) TYPE C VALUE 'Field3',  
\*      fname(10) TYPE C.  
\*  
\*START-OF-SELECTION.  
\*FORMAT COLOR 7.  
\*WRITE : / 'Basic List'.  
\*  
\*  
\*FORMAT HOTSPOT ON.  
\*WRITE : / var1, var2, var3.  
\*FORMAT HOTSPOT off.  
\*  
\*skip 3.  
\*WRITE / 'ust info'.  
\*  
\*AT LINE-SELECTION.  
\*get cursor field fname.  
\*  
\*IF fname = 'VAR1'.  
\*   write : / 'User interacted on Field1'.  
\*elseif fname = 'VAR2'.  
\*   write : / 'User interacted on Field2'.  
\*elseif fname = 'VAR3'.  
\*   write : / 'User interacted on Field3'.  
\*endif.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_GET\_VBAK\_DATA  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_GET\_VBAK\_DATA.  
  
include ZVAISH\_GET\_VBAK\_TOP.  
  
  
SELECT-OPTIONS s\_vbeln for lv\_vbeln.  
\*  
START-OF-SELECTION.  
  
PERFORM get\_vbak\_data.  
PERFORM disp\_vbak\_data.  
  
\*&---------------------------------------------------------------------\*  
\*& Form get\_vbak\_data  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM get\_vbak\_data .  
  
  select VBELN ERDAT ERZET ERNAM ANGDT KUNNR  
    into table lt\_tab  
    from vbak  
    where vbeln in s\_vbeln.  
  
    if sy-subrc eq 0.  
      write: / 'records are found'.  
  
      else .  
        write:/ 'records are not found'.  
        ENDIF.  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*& Form disp\_vbak\_data  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM disp\_vbak\_data .  
  
  loop at lt\_tab into wa\_tab.  
    write:/ wa\_tab-vbeln,  
    wa\_tab-ERDAT,  
    wa\_tab-ERZET,  
    wa\_tab-ERNAM,  
    wa\_tab-ANGDT,  
    wa\_tab-KUNNR.  
    endloop.  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_GET\_VBAK\_FINAL\_DATA  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_GET\_VBAK\_FINAL\_DATA.  
DATA :lt\_tab type zvaish\_TT\_VBAK,  
      wa\_tab type zvaish\_LT\_VBAK.  
select-OPTIONS s\_vbeln for wa\_tab-vbeln.  
  
START-OF-SELECTION.  
CALL FUNCTION 'ZVAISH\_GET\_VBAK\_FINAL'  
EXPORTING  
  I\_VBELN1 = s\_vbeln-low  
  I\_VBELN2 = s\_vbeln-high  
  TABLES  
    gt\_tab = lt\_tab.  
  
 PERFORM display\_get\_vbak.  
\*&---------------------------------------------------------------------\*  
\*& Form display\_get\_vbak  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM display\_get\_vbak .  
  
  loop at lt\_tab into wa\_tab.  
    write:/ wa\_tab-VBELN,  
wa\_tab-ERDAT,  
wa\_tab-ERZET,  
wa\_tab-ERNAM,  
wa\_tab-ANGDT,  
wa\_tab-KUNNR.  
  
    ENDLOOP.  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_INHERITANCE  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_INHERITANCE.  
  
\*class superclass definition.  
\*  public section.  
\*  methods supcmethod1.  
\*  data : supcnum1 type I value 6.  
\*  
\*  PROTECTED SECTION.  
\*  methods supcmethod2.  
\*  data: supcnum2 type i value 10.  
\*  endclass.  
\*  
\*  class superclass implementation.  
\*    method supcmethod1.  
\*      write ' i am public method of super class'.  
\*      endmethod.  
\*    method supcmethod2.  
\*      write 'i am protected method of super class'.  
\*    endmethod.  
\*    endclass.  
\*  
\*  
\*    class subclass DEFINITION inheriting from superclass.  
\*      public section.  
\*      data subc type i value 40.  
\*      methods submethod.  
\*  
\*      endclass.  
\*  
\*      class subclass IMPLEMENTATION.  
\*        method submethod.  
\*              write 'i am public method of subclass'.  
\*              call method supcmethod2. "protected method of superclass can only be accessed in subclass method.  
\*              write supcnum2.  
\*          ENDMETHOD.  
\*        endclass.  
\*START-OF-SELECTION.  
\*\*data supclsobj type REF TO superclass.  
\*\*create object supclsobj.  
\*\*supclsobj->supcmethod1( ).  
\*\*\*supclsobj->supcmethod2( ).  
\*\*write supclsobj->supcnum1.  
\*\*\*write supclsobj->supcnum2.  
\*  
\*data subclsobj type REF TO subclass.  
\*create object subclsobj.  
\*subclsobj->supcmethod1( ).  
\*\*subclsobj->supcmethod2( ).  
\*subclsobj->submethod( ).  
\*write subclsobj->supcnum1.  
\*\*write supclsobj->supcnum2.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*# 2 subclass methods can re-implement inherited methods from superclass  
  
\*class superclass DEFINITION.  
\*  public section.  
\*  methods: supp1.  
\*  protected section.  
\*  methods: supp2.  
\*  endclass.  
\*  
\*  class superclass IMPLEMENTATION.  
\*    method: supp1.  
\*      write 'im superclass method of public visibility'.  
\*      endmethod.  
\*      method supp2.  
\*        write 'im superclass method of private visibility'.  
\*        endmethod.  
\*    ENDCLASS.  
\*  
\*    class subclass DEFINITION INHERITING FROM superclass.  
\*      public section.  
\*      methods supp1 REDEFINITION. "overriding concept from java same method name different application or implementation.  
\*  
\*      PROTECTED SECTION.  
\*        methods supp2 REDEFINITION.  
\*      endclass.  
\*  
\*      class subclass IMPLEMENTATION.  
\*        method supp1.  
\*          write 'i am subclass method of public visibility'.  
\*          endmethod.  
\*          method supp2.  
\*            write 'i am subclass method of protected visibility'.  
\*            endmethod.  
\*        endclass.  
\*        START-OF-SELECTION.  
\*        data :supobj type REF TO superclass,  
\*              subobj type ref to subclass.  
\*  
\*        create object: supobj, subobj.  
\*        supobj->supp1( ). "superclass method  of public visibility  
\*\*supobj->supp2( ). "cannot acces from here as it is protected.  
\*  
\*  
\*\*subobj->subp1( ). "subclass method of public visibility  
\*\*subobj->subp2( ). "caanot as its protected.  
\*        subobj->supp1( ).  
  
  
\*\*\*\*\*\*\*\*\*\*\*#3. objects are not created in abstract class.  
  
\*class superclass DEFINITION ABSTRACT.  
\*  public section.  
\*  methods supmethod1.  
\*  
\*  ENDCLASS.  
\*  
\*\*\*\*\*\*\*\*\*\*\*  class superclass IMPLEMENTATION.  
\*\*\*\*\*\*\*\*\*\*\*    method supmethod1.  
\*\*\*\*\*\*\*\*\*\*\*      write 'abstarct method'.  
\*\*\*\*\*\*\*\*\*\*\*      endmethod.  
\*\*\*\*\*\*\*\*\*\*\*    endclass.  
\*  
\*    class subclass DEFINITION INHERITING FROM superclass.  
\*      public section.  
\*      methods submethod.  
\*  
\*      ENDCLASS.  
\*  
\*\*\*      class subclass IMPLEMENTATION.  
\*\*\*        method submethod.  
\*\*\*          write 'subclass method'.  
\*\*\*          call method supmethod1( ).  
\*\*\*          endmethod.  
\*  
\*        endclass.  
\*        START-OF-SELECTION.  
\*\*data supobj type REF TO superclass.  
\*\*create object supobj. "abstarct class obeject cannot be created.  
\*  
\*        data subobj type REF TO subclass.  
\*        create object subobj.  
\*        subobj->submethod( ).  
\*        subobj->supmethod1( ).  
  
\*\*\*\*\*\*\*\*\*\*# 4  abstract method should be redefined then implemented in\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*class superclass DEFINITION ABSTRACT.  
\*  public section.  
\*  methods supmethod1 abstract.  
\*  methods supmethod2.  
\*  
\*  ENDCLASS.  
\*  
\*  class superclass IMPLEMENTATION.  
\*\*\*\*\*\*\*\*\*    method supmethod1.  "abstract method cannot be implemented until its redifined or overridden.  
\*\*\*\*\*\*\*\*\*      write 'abstarct method'.  
\*\*\*\*\*\*\*\*\*      endmethod.  
\*      method supmethod2.  
\*        write 'non abstarct method'.  
\*        endmethod.  
\*    endclass.  
\*  
\*    class subclass DEFINITION INHERITING FROM superclass.  
\*      public section.  
\*      methods submethod.  
\*  methods supmethod1 REDEFINITION.  
\*      ENDCLASS.  
\*  
\*      class subclass IMPLEMENTATION.  
\*        method submethod.  
\*          write 'subclass method'.  
\*\*call method supmethod1( ).  
\*          endmethod.  
\*          method supmethod1.  
\*            write 'abstarct method of super class'.  
\*            ENDMETHOD.  
\*  
\*        endclass.  
\*        START-OF-SELECTION.  
\*\*data supobj type REF TO superclass.  
\*\*create object supobj. "abstarct class obeject cannot be created.  
\*  
\*        data subobj type REF TO subclass.  
\*        create object subobj.  
\*        subobj->submethod( ).  
\*        subobj->supmethod1( ).  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*5.\*\*\*\*\*\*\*\*\*\*\*  
CLASS superclass DEFINITION FINAL.  
  public section.  
  methods finalmeth.  
ENDCLASS.  
  
CLASS superclass IMPLEMENTATION .  
  method finalmeth.  
    write 'im final method of superclass'.  
    endmethod.  
ENDCLASS.  
  
\*CLASS subclass DEFINITION INHERITING FROM superclass. "final class cannot be inherited  
class subclass DEFINITION.  
  public section.  
  methods submethod.  
ENDCLASS.  
  
CLASS subclass IMPLEMENTATION.  
  method submethod.  
    write 'i am submethod of subclass'.  
    endmethod.  
ENDCLASS.  
  
START-OF-SELECTION.  
DATA : subobj TYPE REF TO subclass.  
CREATE OBJECT subobj.  
subobj->submethod( ).  
\*\*subobj->finalmeth( ). "cannot access as not inherited  
  
data supobj type REF TO superclass.  
create object supobj.  
supobj->finalmeth( ).

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_INLINE\_PROGRAMS  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zvaish\_inline\_programs.  
  
  
  
\*\*\*\*#1.old abap open sql  
  
  
  
\*TYPES: BEGIN OF ls\_tab,  
\*         matnr TYPE matnr,  
\*         mbrsh TYPE mbrsh,  
\*         mtart TYPE mtart,  
\*         meins TYPE meins,  
\*         ntgew TYPE ntgew,  
\*  
\*  
\*       END OF ls\_tab.  
\*  
\*DATA : lt\_tab TYPE STANDARD TABLE OF ls\_tab,  
\*       wa\_tab TYPE ls\_tab.  
\*SELECT-OPTIONS s\_matnr FOR wa\_tab-matnr.  
\*  
\*START-OF-SELECTION.  
\*  
\*  SELECT matnr mbrsh mtart meins ntgew  
\*    INTO TABLE lt\_tab  
\*     FROM mara  
\*     WHERE matnr IN s\_matnr.  
\*  
\*END-OF-SELECTION.  
\*  
\*  LOOP AT lt\_tab INTO wa\_tab.  
\*  
\*    WRITE: / wa\_tab-matnr,  
\*             wa\_tab-mbrsh,  
\*             wa\_tab-mtart,  
\*             wa\_tab-meins,  
\*             wa\_tab-ntgew.  
\*  
\*  ENDLOOP.  
  
  
\*#2. NEW OPEN SQL / INLINE/CODE PUSH DOWN TECHNIQUE  
  
\*  
\*TYPES: BEGIN OF ls\_tab,  "EXCLUDES THIS  
\*         matnr TYPE matnr,  
\*         mbrsh TYPE mbrsh,  
\*         mtart TYPE mtart,  
\*         meins TYPE meins,  
\*         ntgew TYPE ntgew,  
\*  
\*  
\*       END OF ls\_tab.  
\*  
\*DATA : lt\_tab TYPE STANDARD TABLE OF ls\_tab,    wa\_tab TYPE ls\_tab.  
\*  
\*  
\*SELECT-OPTIONS s\_matnr FOR wa\_tab-matnr.  
\*  
\*START-OF-SELECTION.  
\*  
\*  SELECT matnr, mbrsh, mtart, meins, ntgew  
\*    INTO TABLE @lt\_tab  
\*     FROM mara  
\*     WHERE matnr IN @s\_matnr.  
\*  
\*END-OF-SELECTION.  
\*  
\*  LOOP AT lt\_tab INTO wa\_tab.  
\*  
\*    WRITE: / wa\_tab-matnr,  
\*             wa\_tab-mbrsh,  
\*             wa\_tab-mtart,  
\*             wa\_tab-meins,  
\*             wa\_tab-ntgew.  
\*  
\*  ENDLOOP.  
  
  
\*#3, without structure  
  
\*\*\*\*\*\*\*\*\*TYPES: BEGIN OF ls\_tab,  "EXCLUDEd this  
\*\*\*\*\*\*\*\*\*         matnr TYPE matnr,  
\*\*\*\*\*\*\*\*\*         mbrsh TYPE mbrsh,  
\*\*\*\*\*\*\*\*\*         mtart TYPE mtart,  
\*\*\*\*\*\*\*\*\*         meins TYPE meins,  
\*\*\*\*\*\*\*\*\*         ntgew TYPE ntgew,  
\*\*\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*\*       END OF ls\_tab.  
\*\*\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*\*DATA : lt\_tab TYPE STANDARD TABLE OF ls\_tab,  
\*\*\*\*\*\*\*\*\*   wa\_tab TYPE ls\_tab.  
  
  
\*SELECT-OPTIONS S\_MATNR FOR MATNR. "DIRECTLY ASSINGING MATNR WILL NOT GET AS NO STRUCTRE IS REFEERES  
\*DATA LV\_MATNR TYPE MATNR. "THATS WHY LOCAL VARIABLE IS REFERED  
\*  
\*SELECT-OPTIONS s\_matnr FOR LV\_MATNR.  
\*\*  
\*START-OF-SELECTION.  
\*\*  
\* SELECT matnr, mbrsh, mtart, meins, ntgew  
\*    INTO TABLE @DATA(LT\_TAB) "LT\_TAB AT DATABASE LAYER WITH @DATA  
\*     FROM mara  
\*     WHERE matnr IN @s\_matnr.  
\*END-OF-SELECTION.  
\*  
\*  LOOP AT lt\_tab INTO DATA(wa\_tab).  
\*  
\*    WRITE: / wa\_tab-matnr,  
\*             wa\_tab-mbrsh,  
\*             wa\_tab-mtart,  
\*             wa\_tab-meins,  
\*             wa\_tab-ntgew.  
\*  
\*  ENDLOOP.  
  
\*  
\*#4. output display  
  
\*DATA LV\_MATNR TYPE MATNR. "THATS WHY LOCAL VARIABLE IS REFERED  
\*  
\*SELECT-OPTIONS s\_matnr FOR LV\_MATNR.  
\*\*  
\*START-OF-SELECTION.  
\*\*  
\* SELECT matnr, mbrsh, mtart, meins, ntgew  
\*    INTO TABLE @DATA(LT\_TAB) "LT\_TAB AT DATABASE LAYER WITH @DATA  
\*     FROM mara  
\*     WHERE matnr IN @s\_matnr.  
\*END-OF-SELECTION.  
\*  
\* CL\_DEMO\_OUTPUT=>display\_data( lt\_tab ).  
  
  
  
\*#5. alias  
\*DATA lv\_matnr TYPE matnr. "THATS WHY LOCAL VARIABLE IS REFERED  
\*  
\*SELECT-OPTIONS s\_matnr FOR lv\_matnr.  
\*\*  
\*START-OF-SELECTION.  
\*\*  
\*  SELECT matnr AS MaterialNumber,  
\* mbrsh AS MaterIALTYPE,  
\* mtart AS Industry,  
\*meins AS MATERAILPRODUCT,  
\* ntgew AS NETWEIGHT  
\*     INTO TABLE @DATA(lt\_tab) "LT\_TAB AT DATABASE LAYER WITH @DATA  
\*      FROM mara  
\*      WHERE matnr IN @s\_matnr.  
\*  
\*  
\*  cl\_demo\_output=>display\_data( lt\_tab ).  
  
  
\*6. CALCULATIONS  
  
  
\*  
\*DATA lv\_matnr TYPE matnr. "THATS WHY LOCAL VARIABLE IS REFERED  
\*  
\*SELECT-OPTIONS s\_matnr FOR lv\_matnr.  
\*\*  
\*START-OF-SELECTION.  
\*\*  
\*  SELECT  
\*( 8 \* 2 ) AS TWOMULTIPLE,  
\* matnr AS MaterialNumber,  
\* mbrsh AS MaterIALTYPE,  
\* mtart AS Industry,  
\*meins AS MATERAILPRODUCT,  
\* ntgew AS NETWEIGHT  
\*     INTO TABLE @DATA(lt\_tab) "LT\_TAB AT DATABASE LAYER WITH @DATA  
\*      FROM mara  
\*      WHERE matnr IN @s\_matnr.  
\*  
\*  
\*  cl\_demo\_output=>display\_data( lt\_tab ).  
  
  
  
\*#7.  
  
\*DATA lv\_matnr TYPE matnr. "THATS WHY LOCAL VARIABLE IS REFERED  
\*  
\*SELECT-OPTIONS s\_matnr FOR lv\_matnr.  
\*PARAMETERS P\_VAL TYPE NTGEW.  
\*\*  
\*START-OF-SELECTION.  
\*  
\*  SELECT  
\*( 8 \* 2 ) AS TWOMULTIPLE,  
\* matnr AS MaterialNumber,  
\* mbrsh AS MaterIALTYPE,  
\* mtart AS Industry,  
\*meins AS MATERAILPRODUCT,  
\* ntgew AS NETWEIGHT,  
\*NTGEW \* @P\_VAL AS CAL\_NETWEIGHT  
\*     INTO TABLE @DATA(lt\_tab) "LT\_TAB AT DATABASE LAYER WITH @DATA  
\*      FROM mara  
\*      WHERE matnr IN @s\_matnr.  
\*  
\*  
\*  cl\_demo\_output=>display\_data( lt\_tab ).  
  
  
\*\*#9.  
\*  
\*Types TT\_DATA type MD\_RANGE\_T\_MATNR.  
\*  
\*DATA(IT\_TAB9) = NEW TT\_DATA(  
\*( SIGN = 'I' OPTION = 'EQ' LOW = '0030000012')  
\*( SIGN = 'I' OPTION = 'EQ' HIGH = '0030000020')  
\*).  
\*  
\*CL\_DEMO\_OUTPUT=>display\_data( VALUE = IT\_TAB9 ).  
  
  
  
\*\*#12. Inner query  
\*  
\*SELECT vbeln,  
\*  posnr, matnr  
\*  from vbap into table @data(it\_tab)  
\*  where vbeln = any ( select vbeln from vbap  
\*  where vbeln between '0000000002' and '0000000010' ). "first high values get printed andthen lower values  
\*  
\*  CL\_DEMO\_OUTPUT=>display\_data( it\_tab ).  
  
  
  
  
\*#13.  
  
SELECT VBELN, POSNR, MATNR  
  FROM VBAP  
  INTO TABLE @data(lt\_tab)  
  where vbeln BETWEEN '0000000001' and '0000000020'.  
  sort lt\_tab by vbeln DESCENDING.  
  CL\_DEMO\_OUTPUT=>display\_data( lt\_tab ).  
  
  
\*\*#24.1 SWITCH WITH CHAR DATATYPE  
\*  
\*PARAMETERS p\_day TYPE i.  
\*\*\*\*\*\*\*\*\*\*\*data lv\_day type i value 1234. "here we are declaring assing separtely in different lines  
\*\*\*\*\*\*\*\*\*\*\*write lv\_day.  
\*  
\*DATA(lv\_day) = SWITCH char7( p\_day "MORE THAN 7 IF LETTERS ARE THERE THEY ARE NO PRINTED EX:WEDNESDAY  
\*                  WHEN 1 THEN 'MONDAY'  
\*                  WHEN 2 THEN 'TUESDAY'  
\*                  WHEN 3 THEN 'WEDNESDAY'  
\*                  WHEN 4 THEN 'THURSDAY'  
\*                  WHEN 5 THEN 'FRIDAY'  
\*                  WHEN 6 THEN 'SATURDAY'  
\*                  WHEN 7 THEN 'SUNDAY'  
\*                  ELSE 'INVALID' ).  
\*  
\*WRITE :/ p\_day, lv\_day  COLOR 4. "COLOR 4 IS FOR HIGHLIGHTER OR BACKGROUND COLOR.  
  
\*#24.2 SWITCH # "TAKES LENGTH OF FIRST STATMENT  
  
\*PARAMETERS P\_DAY TYPE I.  
\*  
\*DATA(LV\_DAY) = SWITCH #( P\_DAY       "# TAKES THE LENGTH OF THE FIRST STMT IN SWITCH  
\*WHEN 1 THEN 'MONDAY'  
\*WHEN 2 THEN 'TUESDAY'  
\*WHEN 3 THEN 'WEDNESDAY'  
\*WHEN 4 THEN 'THURSDAY'  
\*WHEN 5 THEN 'FRIDAY'  
\*WHEN 6 THEN 'SATURDAY'  
\*WHEN 7 THEN 'SUNDAY'  
\*ELSE 'INVALID'  
\*).  
\*WRITE : / P\_DAY , LV\_DAY COLOR 4.  
  
  
\*#15  
  
\*  
"normal ABAP PROGRAM" "static length"  
\*DATA LV\_TEXT(10) TYPE C VALUE 'ABAP ON HANA UST'.  
\*WRITE: LV\_TEXT. "BASED ON LENGTH IT GETS PRINTED"  
\*  
\*  
\*  
\*"INLINE PROGRAM" "dynamic length"  
\*DATA(LV\_TEXTIN) = 'ABAP ON HANA MB INLINE CODE HELLO '.  
\*WRITE : LV\_TEXTIN. "Length specification is not  required  
\*  
\*data(lv\_text1) = | ABAP | & | | & | ON | & | HANA UST ABAP CODE|. "SPACES SHOULD BE THERE BETWEEN PIPLINES AND &  
\*WRITE / LV\_TEXT1.  
\*  
\*CL\_DEMO\_OUTPUT=>display\_data(  
\*value = lv\_text1  
\*\*include =  
\*name = 'ust\_info').  
\*  
\*  
  
  
  
\*#8.1  
  
\*TYPES LS\_TAB TYPE STANDARD TABLE OF I WITH DEFAULT KEY.  
\*  
\*DATA(it\_matnr) = value ls\_tab( ( 234 ) ( ) ( 2340 ) ). "printed as it is  
\*  
\*CL\_DEMO\_OUTPUT=>display\_data( it\_matnr ). "print like a table with it\_matnr as coloumn name or heading of table.  
\*  
\*TYPES LS\_TAB1 TYPE sorted TABLE OF I WITh unique key table\_line. "sorted table so unique key  
\* data(it\_matnr1) = value ls\_tab1( ( 100 )  ( 20 ) ( 30 ) ). "printed after sorting  
\* "20  
\* "30  
\* "100  
\* CL\_DEMO\_OUTPUT=>display\_data( it\_matnr1 ).  
\*\* WRITE it\_matnr1. "cannot be converted as character like value  
\*  
\*  
\*\*#8.2  
  
\*types ls\_tab type STANDARD TABLE OF I with default key.  
\*  
\*DATA(IT\_MATNR) = NEW ls\_tab( ( 234 ) ( ) ( 2345 ) ). "IT\_MATNR IS OBJECT  
\*  
\*CL\_DEMO\_OUTPUT=>display( IT\_MATNR ).  
\*CL\_DEMO\_OUTPUT=>display( IT\_MATNR->\* ).  
\*ASSIGN IT\_MATNR->\* TO FIELD-SYMBOL(<FS>).  
\*CL\_DEMO\_OUTPUT=>display( <FS> ).  
  
\*#18.1  
  
\*DATA : F1(20) TYPE C.  
\*F1 = 'UST GLOBAL'.  
\*  
\*FIELD-SYMBOLS <FS>. "POINTER  
\*  
\*ASSIGN F1 TO <FS>.  
\*WRITE :/ F1. "UST GLOBAL  
\*WRITE : <FS>. "UST GLOBAL  
  
  
\*#18.2  
  
  
  
\*#20  
\*TABLES VBAP.  
\*  
\*PARAMETERS P\_VBELN TYPE VBELN\_VA.  
\*DATA IT\_TAB1 TYPE STANDARD TABLE OF vbap.  
\*data IT\_TAB2 type STANDARD TABLE OF vbap with NON-UNIQUE SORTED KEY VBELN COMPONENTS VBELN.  
\*  
\*SELECT \* FROM VBAP  
\*  INTO TABLE @IT\_TAB2.  
\*  IF SY-SUBRC EQ 0.  
\*  
\*    IT\_TAB1 = FILTER #( IT\_TAB2 USING KEY VBELN WHERE VBELN = P\_VBELN ).  
\*    ELSE.  
\*      WRITE 'NO RECORDS'.  
\*      ENDIF.  
\*  
\*      CL\_DEMO\_OUTPUT=>display\_data( IT\_TAB1 ).  
  
  
\*#21.  
  
\*TABLES: VBAK, VBAP.  
\*  
\*SELECT-OPTIONS S\_VBELN FOR VBAK-VBELN.  
\*  
\*SELECT SO\_H~\*,  
\*  SO\_I~VBELN,  
\*  SO\_I~POSNR,  
\*  SO\_I~MATNR,  
\*  SO\_I~NETWR,  
\*  SO\_I~MATWA,  
\*  SO\_I~NETPR  
\*  
\*  
\*  INTO TABLE @DATA(IT\_SALES)  
\*  FROM VBAK AS SO\_H  
\*  INNER JOIN VBAP AS SO\_I  
\*  ON SO\_H~VBELN = SO\_I~VBELN  
\*  WHERE SO\_H~VBELN IN @S\_VBELN.  
\*  
\*  CL\_DEMO\_OUTPUT=>display\_data( IT\_SALES ).  
  
  
\* #22.  
\*  
\*TYPES : BEGIN OF ls\_tab,  
\*   kunnr type kunnr,  
\*  name1 type name1,  
\*  name2 type name2,  
\*  ort01 type ort01,  
\*   t\_col type lvc\_t\_scol,  
\*  
\*  end of ls\_tab.  
\*  
\*  types: ls\_tab2 type STANDARD TABLE OF ls\_tab with DEFAULT KEY. "if this not defined gives error  
\*  
\*  data(it\_tab) = new ls\_tab2( (  
\*  
\*  kunnr = '1234'  
\*  name1 = 'abc'  
\*  name2 = 'ust info tech'  
\*  ort01 = 'hyd'  
\*  
\*  t\_col = VALUE #( (  
\*  FNAME = 'COL1' COLOR-COL = 2 COLOR-INV = 1 )  
\*  ( FNAME = 'COL2' COLOR-COL = 2 COLOR-INV = 1 )  
\*  ) ) ).  
\*  
\*  CL\_DEMO\_OUTPUT=>display( it\_tab ).  
  
  
  
\*#18.2  
\*DATA: BEGIN OF line, "field string  
\* col\_1 type i,  
\*col\_2 type i,  
\*end of line.  
\*  
\*DATA it\_tab like sorted table of line with UNIQUE key col\_1.  
\*FIELD-SYMBOLS <fs> like line of it\_tab.  
\* DO 10 TIMES.  
\*  
\*line-col\_1 = sy-index.  
\*line-col\_2 = 2 \* sy-index.  
\*  
\*Append line to it\_tab.  
\*enddo.  
\*  
\*READ TABLE it\_tab ASSIGNING <fs> index 6.  
\*  
\*write: sy-subrc, sy-tabix.  
\* write:/ <fs>-col\_1, <fs>-col\_2.  
  
\*#18.3 HANA  
\*  
\*TYPES: BEGIN OF ls\_tab,  
\*  vbeln type vbeln\_va,  
\*  posnr type posnr\_va,  
\*  end of ls\_tab.  
\*   DATA: it\_tab type STANDARD TABLE OF ls\_tab,  
\*         wa\_tab type ls\_tab.  
\*  
\*   it\_tab = VALUE #(  (  VBELN = '0030000010' POSNR = '000010' )  
\*( VBELN = '0030000020' POSNR = '000010' ) ).  
\*  
\*   LOOP AT IT\_TAB ASSIGNING FIELD-SYMBOL(<FS\_VBAP>).  
\*  
\*     CL\_DEMO\_OUTPUT=>display\_data( VALUE = <FS\_VBAP> ).  
\*  
\*     ENDLOOP.  
  
\*#19  
  
\*DATA LV\_POSNR TYPE POSNR\_VA.  
\*DATA LV\_MATNR TYPE MATNR.  
\*SELECT VBELN,  
\*  POSNR,  
\*  MATNR  
\*  FROM VBAP  
\*  INTO TABLE @DATA(IT\_TAB).  
\*  WHERE VBELN IN S\_VBELN.  
\*  
\*  READ TABLE IT\_TAB.  
\*  
\*  IF SY-SUBRC EQ 0.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_INTABLE  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zvaish\_intable.  
  
\*\*\*\*without itab standard table  
  
  
\*DATA : lv\_kunnr type kunnr,  
\*      lv\_land1 type LAND1\_GP,  
\*      lv\_NAME1 TYPE NAME1\_GP,  
\*      lv\_ort01 type ort01\_gp.  
\*  
\*  
\*select-options s\_kunnr for lv\_kunnr.  
\*PARAMETERS p\_land1 type kna1-land1.  
\*  
\*select kunnr name1 land1 ort01  
\*  into ( lv\_kunnr,lv\_name1,lv\_land1,lv\_ort01 )  
\*  from kna1  
\*  where kunnr in s\_kunnr  
\*  and land1 eq p\_land1.  
\*  write :/ lv\_kunnr,lv\_name1,lv\_land1, lv\_ort01.  
\*  ENDSELECT.  
\*  
  
\*\*\*\*\*\*\*\*\*\*without itab emp table  
  
DATA : LV\_emp\_id     TYPE  zemp\_id,  
       LV\_emp\_name   TYPE  zemp\_name,  
       LV\_emp\_address TYPE  zemp\_address,  
       LV\_emp\_phone  TYPE  zemp\_phone.  
  
SELECT-OPTIONS s\_EMPID FOR LV\_EMP\_ID.  
PARAMETERS p\_Name TYPE zvaish\_emp\_table-emp\_name.  
  
SELECT EMP\_ID EMP\_NAME EMP\_ADDRESS EMP\_PHONE  
  INTO ( lv\_emp\_id,lv\_emp\_name,lv\_emp\_address,lv\_emp\_phone )  
  FROM zvaish\_emp\_table  
  WHERE emp\_id IN s\_empid  
  AND emp\_name EQ p\_name.  
  WRITE :/ lv\_emp\_id,lv\_emp\_name,lv\_emp\_address, lv\_emp\_phone.  
ENDSELECT.  
  
  
  
  
\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*without itab standard table  
  
\*  
\*  DATA: lv\_kunnr type kunnr,  
\*        lv\_name1 type name1\_gp,  
\*        lv\_land1 type land1\_gp,  
\*        lv\_ort01 type ort01\_gp.  
\*  
\*  SELECT-OPTIONS s\_kunnr for lv\_kunnr.  
\*  PARAMETERS p\_land1 type kna1-land1.  
\*  
\*  SELECT KUNNR NAME1 LAND1 ORT01  
\*    INTO ( lv\_kunnr,lv\_name1,lv\_land1,lv\_ort01 )  
\*    from kna1  
\*    where kunnr in s\_kunnr  
\*    and land1 eq p\_land1.  
\*write : / lv\_kunnr,lv\_name1,lv\_land1,lv\_ort01.  
\*    ENDSELECT.  
  
  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*without itab candidate table  
\*  
\*DATA:  
\*lv\_CDTID  type ZCDT\_ID,  
\*lv\_CDTNAME   type ZCDT\_NAME,  
\*lv\_CDTPHONE  type ZCDT\_PHONE,  
\*lv\_CDTCITY  type ZCDT\_CITY.  
\*  
\*select-OPTIONS s\_cdit for lv\_cdtid.  
\*PARAMETERS p\_city type zcandidate-cdt\_city.  
\*  
\*select CDT\_ID CDT\_NAME CDT\_PHONE CDT\_CITY  
\*  into ( lv\_cdtid,lv\_cdtname,lv\_cdtphone, lv\_cdtcity )  
\*  from ZCANDIDATE  
\*  where cdt\_id in s\_cdit  
\*  and cdt\_city eq p\_city.  
\*  write:/ lv\_cdtid, lv\_cdtname, lv\_cdtphone, lv\_cdtcity.  
\*  ENDSELECT.  
\*  
  
\*  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*with itab with header standard table  
  
\*DATA: it\_tab type kna1 occurs 0 WITH HEADER LINE.  
\*data: lv\_kunnr type kunnr.  
\*SELECT-OPTIONS s\_kunnr for lv\_kunnr.  
\*PARAMETERS p\_land1 type kna1-land1.  
\*  
\*SELECT \*  
\*  into it\_tab  
\*  from kna1  
\*  where kunnr in s\_kunnr  
\*  and land1 eq p\_land1.  
\*  
\*  append it\_tab to it\_tab.  
\*  ENDSELECT.  
\*  loop at it\_tab into it\_tab.  
\*    write :/ it\_tab-kunnr,  
\*    it\_tab-name1,  
\*    it\_tab-land1,  
\*    it\_tab-ort01.  
\*    ENDLOOP.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*with header candidate table  
  
\*data: it\_tab type zcandidate occurs 0 WITH HEADER LINE.  
\*data lv\_cdtid type zcdt\_id.  
\*SELECT-OPTIONS s\_cdtid for lv\_cdtid.  
\*PARAMETERS p\_city type zcandidate-cdt\_city.  
\*select \*  
\*  into it\_tab  
\*  from zcandidate  
\*  where cdt\_id in s\_cdtid  
\*  and cdt\_city eq p\_city.  
\*  append it\_tab to it\_tab.  
\*  ENDSELECT.  
\*  
\*  loop  at it\_tab into it\_tab.  
\*    write:/ it\_tab-cdt\_id,  
\*    it\_tab-cdt\_name,  
\*    it\_tab-cdt\_city.  
\*    ENDLOOP.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*without header  
  
\*data:it\_tab type zcandidate occurs 0 WITH HEADER LINE.  
\*data : lv\_cdtid type zcdt\_id.  
\*SELECT-OPTIONS s\_cdtid for lv\_cdtid.  
\*PARAMETERS p\_city type zcandidate-cdt\_city.  
\*  
\*select \*  
\*  into table it\_tab  
\*  from zcandidate  
\*  where cdt\_id in s\_cdtid  
\*  and cdt\_city eq p\_city.  
\*  
\*  
\*  
\*  loop at it\_tab into it\_tab.  
\*    write:/ it\_tab-cdt\_id,  
\*    it\_tab-cdt\_name,  
\*    it\_tab-cdt\_city.  
\*    ENDLOOP.  
\*  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*userdefined workarea  
\*  
\*types: BEGIN OF ls\_tab,  
\*  lv\_cdtid type zcdt\_id,  
\*  lv\_cdtname type zcdt\_name,  
\*  lv\_cdtcity type zcdt\_city,  
\*  end of ls\_tab.  
\*  
\*  data:lv\_cdtid type zcdt\_id.  
\*  SELECT-OPTIONS s\_cdtid for lv\_cdtid.  
\*  PARAMETERS p\_city type zcandidate-cdt\_city.  
\*  
\*  data:it\_tab type  standard table of ls\_tab,  
\*  wa\_tab type ls\_tab.  
\*  
\*  select cdt\_id cdt\_name cdt\_city  
\*    into wa\_tab  
\*    from zcandidate  
\*    where cdt\_id in s\_cdtid  
\*    and cdt\_city eq p\_city.  
\*  
\*    append wa\_tab to it\_tab.  
\*    ENDSELECT.  
\*  
\*    loop at it\_tab into wa\_tab.  
\*      write:/ wa\_tab-lv\_cdtid,  
\*      wa\_tab-lv\_cdtname,  
\*      wa\_tab-lv\_cdtcity.  
\*      ENDLOOP.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_INTERACTIVE\_ATUSERCMD  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zvaish\_interactive\_atusercmd NO STANDARD PAGE HEADING LINE-COUNT 20(3)  
LINE-SIZE 500 MESSAGE-ID zvaish\_messages.  
  
LOAD-OF-PROGRAM.  
  INCLUDE zvaish\_cmd\_top.  
  
  SELECT-OPTIONS s\_matnr FOR wa\_tab-matnr OBLIGATORY.  
  
INITIALIZATION.  
  CLEAR: wa\_tab, lt\_tab.  
  s\_matnr-low = '1'.  
  s\_matnr-high = '100'.  
  APPEND s\_matnr.  
  
AT SELECTION-SCREEN ON s\_matnr.  
  
  SELECT SINGLE matnr  
    INTO lv\_matnr  
    FROM mara  
    WHERE matnr IN s\_matnr.  
  
  IF sy-subrc EQ 0.  
    MESSAGE  s000.  
  ELSE.  
    MESSAGE e001.  
  ENDIF.  
  
START-OF-SELECTION.  
  
  SET PF-STATUS 'ZVAISH\_PFS'.  
  
  PERFORM get\_mat\_data.  
  
END-OF-SELECTION.  
  
  PERFORM disp\_mat\_data.  
  
  DATA lv\_count TYPE i.  
  lv\_count = sy-linct - sy-linno.  
  SKIP lv\_count.  
  
  
TOP-OF-PAGE.  
  ULINE.  
  WRITE :/ 'Records are found in marc table','the current list index is: ', sy-lsind.  
  WRITE :/ 'Material No',  
  30 'Mat Ind',  
  40 'Mat type',  
  55 'Mat Uom',  
  70 'Gross weight',  
  90 'Net Weight'.  
  ULINE.  
  
END-OF-PAGE.  
  
  WRITE :/'current list page no:' , sy-pagno,  
  'Date :' , sy-datum,  
  'time:', sy-uzeit.  
  
AT USER-COMMAND.  
  CASE sy-ucomm.  
    WHEN 'DISPLAY'.  
      WRITE 'UST DISPLAY'.  
    WHEN 'INFO'.  
      WRITE 'info button'.  
    WHEN 'TCODE'.  
      CALL TRANSACTION 'ZTRUPEN\_UST\_MATMAST'.  
    WHEN 'SAP'.  
      WRITE 'SAP LABS BNGL'.  
  
  ENDCASE.  
  
  
  
FORM get\_mat\_data .  
  SELECT matnr mbrsh mtart meins brgew ntgew  
    INTO TABLE lt\_tab  
    FROM mara  
  WHERE matnr IN s\_matnr.  
  IF sy-subrc EQ 0.  
    WRITE:/'Records are found in mara table'.  
  ELSE.  
    WRITE:/ 'Records are not found in mara table'.  
  ENDIF.  
  
ENDFORM.  
  
FORM disp\_mat\_data .  
  
  LOOP AT lt\_tab INTO wa\_tab.  
    WRITE:/ wa\_tab-matnr hotspot,  
    30 wa\_tab-mbrsh,  
    40 wa\_tab-mtart,  
    55 wa\_tab-meins,  
    70 wa\_tab-brgew,  
    90 wa\_tab-ntgew.  
  
  
  
  
  
 hide wa\_tab-matnr.  
  ENDLOOP.  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_INTERACTIVE\_REPORT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_INTERACTIVE\_REPORT NO STANDARD PAGE HEADING LINE-COUNT 20(3).  
  
load-OF-PROGRAM.  
  
include ZVAISH\_GET\_VBAK\_TOP.  
select-options  s\_vbeln for wa\_tab-vbeln.  
  
  initialization.  
 clear : lt\_tab , wa\_tab.  
 s\_vbeln-low = '1'.  
 s\_vbeln-high = '50'.  
 append s\_vbeln.  
  
  at SELECTION-SCREEN on s\_vbeln.  
    select single vbeln  
      into lv\_vbeln  
      from vbak  
      where vbeln in s\_vbeln.  
  
      if sy-subrc eq 0.  
        MESSAGE s000(zvaish\_messages).  
        else.  
          message e001(zvaish\_messages).  
        ENDIF.  
  
  
  START-OF-SELECTION.  
  
  PERFORM get\_vbak\_data.  
  
  
  PERFORM disp\_vbak\_data.  
  
  TOP-OF-PAGE.  
  uline.  
  write:/  
  
  'Sales Document',  
10 'Record Created On',  
20 'Entry time',  
30 'Name of Person Responsible for Creating the Object',  
40 'Quotation/Inquiry is Valid From',  
50 'Sold-to Party'.  
  
  uline.  
  END-OF-PAGE.  
  write:/ 'date', sy-datum,  
  20 'time', sy-uzeit.  
  END-OF-SELECTION.  
  
  AT LINE-SELECTION.  
    CASE sy-lsind.  
      WHEN 1.  
        select KUNNR LAND1 NAME1 NAME2 ORT01  
          into  table lt\_tab2  
          from kna1  
          where kunnr eq wa\_tab-kunnr.  
  
          loop at lt\_tab2 into wa\_tab2.  
            write : /3 wa\_tab2-KUNNR HOTSPOT,  "exactly after 3 slash should be given  
            30 wa\_tab2-land1,  
            40 wa\_tab2-name1,  
            55 wa\_tab2-name2,  
            60 wa\_tab2-ort01.  
  
            hide wa\_tab2-kunnr.  
            ENDLOOP.  
  
  
    ENDCASE.  
  
FORM get\_vbak\_data .  
  
  select VBELN ERDAT ERZET ERNAM ANGDT kunnr  
    into table lt\_tab  
    from vbak  
    where vbeln between s\_vbeln-low and s\_vbeln-high.  
  
ENDFORM.  
  
FORM disp\_vbak\_data .  
  loop at lt\_tab into wa\_tab.  
        write :/ wa\_tab-VBELN  HOTSPOT,  
wa\_tab-ERDAT,  
wa\_tab-ERZET,  
wa\_tab-ERNAM,  
wa\_tab-ANGDT,  
wa\_tab-kunnr.  
    ENDLOOP.  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_INTERFACES  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_INTERFACES.  
\*\*\*#1.  
\*INTERFACE i1.  "objets cannot be created for interfaces  
\*\*data : num type i value 5. "only declaration or structure in interface not values  
\*  data : num type i.  
\*  methods : m1.  
\*  ENDINTERFACE.  
\*  
\*  class c1 definition.  
\*    public section.  
\*    methods : m1.  
\*    interfaces : i1. "inbuilting interface in class as interface is implemented in java  
\*    endclass.  
\*  
\*    class c1 IMPLEMENTATION.  
\*  
\*      method i1~m1. "implementing interface method in class.  
\*        write :/ 'i am mehod 1 m1 from interface'.  
\*        endmethod.  
\*  
\*        method m1.  
\*          write : 'i am method 1 m1 of class'.  
\*          endmethod.  
\*  
\*  
\*      ENDCLASS.  
\*  
\*      START-OF-SELECTION.  
\*  
\*      data iobj type REF TO c1.  
\*      create object iobj.  
\*      iobj->m1( ).  
\*      iobj->i1~m1( ).  
\*write iobj->i1~num.  
  
\*#2.  
\*INTERFACE I2.  
\*  DATA: NUM TYPE I.  
\*  methods : m1,m2.  
\*  ENDINTERFACE.  
\*  
\*  class c2 DEFINITION.  
\*    public section.  
\*    methods : cm2.  
\*      interfaces i2.  
\*    endclass.  
\*  
\*    class c2 IMPLEMENTATION.  
\*      method cm2.  
\*        write /'i am method 2 of class 2'.  
\*        ENDMETHOD.  
\*  
\*        method i2~m2.  
\*          write 'i am method 2 of interface 2'.  
\*          endmethod.  
\*  
\*          method i2~m1.  
\*            write:/ 'i am method 1 of interface 2'.  
\*            ENDMETHOD.  
\*  
\*      endclass.  
\*  
\*      START-OF-SELECTION.  
\*      data iobj2 type REF TO c2.  
\*      create object iobj2.  
\*      iobj2->cm2( ).  
\*         iobj2->i2~m1( ).  
\*      iobj2->i2~m2( ).  
  
\*\*\*\*\*\*\*\*\*\*\*#3.  
\*interface i1.  
\*  data: num1 type I ,  
\*        num2 type I.  
\*  ENDINTERFACE.  
\*  
\*  class c1 definition.  
\*    public section.  
\*    interfaces i1 DATA VALUES num1 = 10 num2 = 20. "assing values to the data  
\*    methods m1.  
\*    endclass.  
\*  
\*    class c1 IMPLEMENTATION.  
\*      method m1.  
\*        write  : / 'num1 value of interface is ',i1~num1.  
\*        write :/'num2 value of interface is',i1~num2.  
\*        ENDMETHOD.  
\*  
\*      ENDCLASS.  
\*  
\*      START-OF-SELECTION.  
\*      data iobj type REF TO c1.  
\*      create object iobj.  
\*      iobj->m1( ).  
  
  
\*\*\*\*\*\*\*\*\*\*\*#4  
  
\*interface i1.  
\*  methods :m1, m2.  
\*  ENDINTERFACE.  
\*  
\*  class superclass definition.  
\*    public section.  
\*    interfaces i1 final methods m2. "assing m2 of interface as final method  
\*    endclass.  
\*  
\*    class superclass IMPLEMENTATION.  
\*        method i1~m1.  
\*      write: /'i am interfac method1 in superclass'.  
\*    endmethod.  
\*  
\*    method i1~m2.  
\*       write: /'i am interfac method2 in superclass'.  
\*      ENDMETHOD.  
\*  
\*  
\*      ENDCLASS.  
\*  
\*      class subclass DEFINITION INHERITING FROM superclass.  
\*        public section.  
\*        methods: i1~m1 REDEFINITION.  
\*   methods: i1~m2 REDEFINITION. "as m2 is final method in superclas it cannot be redefined or implemeted in subclass.  
\*  
\*        endclass.  
\*        class subclass IMPLEMENTATION.  
\*          method i1~m1.  
\*            write :/ 'i am interface method 1 in subclass inherited from superclass'.  
\*            endmethod.  
\*          endclass.  
\*  
\*      START-OF-SELECTION.  
\*      data iobj1 type REF TO superclass.  
\*      data iobj2 type ref to subclass.  
\*      create object : iobj1, iobj2.  
\*      iobj1->i1~m1( ). "method1 in superclass  
\*      iobj1->i1~m2( ).  "method2 in superclass  
\*      iobj2->i1~m1( ). "method1 in subclass  
\*      iobj2->i1~m2( ). " method2 in superclass.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#5.  
\*interface i1.  
\*  data: num type i.  
\*  methods :m1, m2.  
\*  ENDINTERFACE.  
\*  
\*  class superclass definition abstract.  
\*    public section.  
\*    interfaces: i1 ABSTRACT METHODS m2. "assing m2 of interface as abstract method  
\*    endclass.  
\*  
\*    class superclass IMPLEMENTATION.  
\*      method i1~m1.  
\*      write: /'i am interface method1 in superclass'. "it is not abstract so it can be implemented here itself.  
\*    endmethod.  
\*  
\*  
\*  
\*  
\*      ENDCLASS.  
\*  
\*      class subclass DEFINITION INHERITING FROM superclass.  
\*        public section.  
\*  
\*        methods: i1~m2 REDEFINITION. "as m2 is abstract method in superclas it cannot be redefined or implemeted in subclass.  
\*methods : i1~m1 REDEFINITION.  
\*        endclass.  
\*        class subclass IMPLEMENTATION.  
\*             method i1~m1.  
\*  .        super->i1~m1( ).  
\*              write:/ ' i am interface method 1 in subclass inherited'.  
\*             ENDMETHOD.  
\*  
\*            method i1~m2.  
\*              write:/ ' i am interface method 2 in subclass inherited'.  
\*             ENDMETHOD.  
\*          endclass.  
\*  
\*      START-OF-SELECTION.  
\*      data iobj1 type REF TO superclass.  
\*      data iobj2 type ref to subclass.  
\*\*      create object iobj1. "object cannot be created for abstract class.  
\*      create object :  iobj2.  
\*  
\*      iobj2->i1~m1( ). "method1 in subclass  
\*      iobj2->i1~m2( ). " method2 in superclass.  
\*\*      super->i1~m1( ). "cannot be used here only used in instance methods  
  
  
\*\*\*#6\*\*\*\*\*\*\*\*\*  
  
interface i1.  
\*  data: num type i value 10. "cannot assign value here.  
  constants : c\_name(20) type c value 'SAP ABAP'. "but constants can be given value here only  
data num type i .  
class-data cnum type i.  
class-methods : sm.  
methods: im.  
  ENDINTERFACE.  
  
  class superclass definition .  
    public section.  
    interfaces: i1 data values num = 10 cnum = 20. "assing both variables at once no change both with data values  
  
    endclass.  
  
    class superclass IMPLEMENTATION.  
      method i1~im.  
      write: /'i am interface instance method1 in superclass'.  
    endmethod.  
  
    method i1~sm.  
      write :/ 'i am static interface method in superclass'.  
      endmethod.  
  
      ENDCLASS.  
START-OF-SELECTION.  
data: supobj type REF TO superclass,  
 iobj type ref to i1.  
create object supobj.  
\*create object iobj."object cannot be create dfor interface  
write: supobj->i1~c\_name,"sap abap  
 superclass=>i1~cnum,  
 supobj->i1~num.  
supobj->i1~im( ).  
supobj->i1~sm( ).  
superclass=>i1~sm( ).

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_ITAB\_PROGRAMS  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_ITAB\_PROGRAMS.  
\*  
\* DATA : BEGIN OF LINE,   "line is same like work area  
\*   col\_1(10) type c,  
\*   col\_2(20) type c,  
\*   END OF LINE.  
\*  
\*   data itab like TABLE OF line.  
\*  
\* line-col\_1 = 'ust'.  
\*   line-col\_2 = 'global'.  
\*   append line to itab.  
\*   clear line.  
\*   refresh itab.  
\*    if itab is initial.  
\*      write 'internal table is empty'.  
\*  
\*      ENDIF.  
  
  
\*\*\*\*\*\*\*\*insert  
\*  
\*data : BEGIN OF line,  
\*name(10) type c,  
\*  age type i,  
\*  weight(10) type p decimals 2 ,"p means packaed 2 means 2 decimals 1.23  
\* land(10) type c,  
\*  end of line.  
\*  
\*  types: begin of itab,  
\*   name type c,  
\*  age type i,  
\*  weight type p decimals 2 ,"p means packaed 2 means 2 decimals 1.23  
\* land(5) type c,  
\*  end of itab.  
\*  
\*data itab like  table of line with non-unique key name age weight land.  
\*data itab like SORTED TABLE OF line with non-unique key name age weight land.  
\*  
\* line-name = 'edwin'.  
\* line-age = '21'.  
\* line-weight = '23.78'.  
\* line-land = 'denmark'.  
\* insert line into table itab.  
\*  
\*  line-name = 'mark'.  
\* line-age = '24'.  
\* line-weight = '53.78'.  
\* line-land = 'greek'.  
\* insert line into table itab.  
\*  
\*  line-name = 'john'.  
\* line-age = '34'.  
\* line-weight = '63.78'.  
\* line-land = 'europe'.  
\* insert line into table itab.  
\*  
\* loop at itab into line.  
\*   write: /  
\*   line-name,  
\*   line-age,  
\*   line-weight,  
\*   line-land.  
\*   ENDLOOP.  
  
data :BEGIN OF line,  
col\_1 type i,  
  col\_2 type i,  
  end of line.  
  
  data lt\_tab like TABLE of line.  
  
  do 2 times.      "entering data into itab from line  
    line-col\_1 = sy-index.  
    line-col\_2 = sy-index \*\* 2.  
    APPEND line to lt\_tab.  
    ENDDO.  
  
\*    loop at lt\_tab into line.  " looping data from it\_tab into line  
\*  
\*    line-col\_1 = 3 \* sy-tabix.  
\*    line-col\_2 = 5 \* sy-tabix.  
\*      endloop.  
 loop at lt\_tab into line.  " looping data from it\_tab into line  
  
    line-col\_1 = 3 \* sy-tabix.  
    line-col\_2 = 5 \* sy-tabix.  
    insert line into lt\_tab.  
      endloop.  
  
      loop at lt\_tab into line.  
        write:/ sy-tabix, line-col\_1, line-col\_2.  
        ENDLOOP.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_JOIN  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_JOIN.  
\*&---------------------------------------------------------------------\*  
  
\*  
\*\*\*\*\*\*# 1  
\*TYPES: BEGIN OF ls\_tab,  
\*         vbeln TYPE vbeln\_va,  
\*         audat TYPE audat,  
\*         vbtyp TYPE vbtypl,  
\*         trvog TYPE trvog,  
\*         auart TYPE auart,  
\*       END OF ls\_tab.  
\*  
\*DATA : lt\_tab TYPE STANDARD TABLE OF ls\_tab,  
\*       wa\_tab TYPE ls\_tab.  
\*  
\*SELECT-OPTIONS s\_vbeln FOR wa\_tab-vbeln.  
\*  
\*START-OF-SELECTION.  
\*  
\*  SELECT vbeln audat vbtyp trvog auart  
\*    INTO TABLE lt\_tab  
\*    FROM vbak  
\*    WHERE vbeln IN s\_vbeln.  
\*  
\*  IF sy-subrc EQ 0.  
\*    WRITE : 'sales order info ', sy-dbcnt.  
\*  ELSE.  
\*    WRITE : 'sales order records are not found ', sy-dbcnt.  
\*  ENDIF.  
\*  
\*END-OF-SELECTION.  
\*  LOOP AT lt\_tab INTO wa\_tab.  
\*    WRITE : / wa\_tab-vbeln,  
\*    wa\_tab-audat,  
\*    wa\_tab-vbtyp,  
\*    wa\_tab-trvog,  
\*    wa\_tab-auart.  
\*  ENDLOOP.  
  
  
  
\*\*\*\*\*\*# 2 two tables  
  
TYPES : BEGIN OF LS\_TAB,  
KUNNR  TYPE KUNNR,  
LAND1  TYPE LAND1\_GP,  
NAME1  TYPE NAME1\_GP,  
  
VBELN  TYPE VBELN\_VA,  
ERDAT   TYPE ERDAT,  
ERZET  TYPE ERZET,  
  
END OF LS\_TAB.  
  
DATA: wa\_tab type ls\_tab,  
      lt\_tab type table of ls\_tab,  
      lv\_kunnr type kunnr.  
  
SELECT-OPTIONS S\_KUNNR FOR LV\_KUNNR.  
  
select  
  kna1~kunnr  
  kna1~land1  
  kna1~name1  
  
  vbak~vbeln  
  vbak~erdat  
  vbak~erzet  
  
   into table lt\_tab  
   from kna1  
  inner JOIN vbak ON kna1~kunnr = vbak~kunnr.  
  
  if sy-subrc eq 0.  
    write: 'records found'.  
    else.  
      write: 'no records found'.  
    endif.  
  
    loop at lt\_tab into wa\_tab.  
      write: wa\_tab-vbeln,  
      wa\_tab-kunnr,  
      wa\_tab-erdat,  
      wa\_tab-land1,  
      wa\_tab-name1.  
  
     ENDLOOP.  
\*  
\*\*\*\*\*\*# 2 two tables with alias  
\*TYPES: BEGIN OF ls\_tab,  
\*         vbeln TYPE vbeln\_va,  "  vbak  
\*         audat TYPE audat,  
\*         vbtyp TYPE vbtypl,  
\*         trvog TYPE trvog,  
\*         auart TYPE auart,  
\*  
\*         posnr TYPE posnr\_va,  "   vbap  
\*         matnr TYPE matnr,  
\*         matwa TYPE matwa,  
\*       END OF ls\_tab.  
\*  
\*DATA : lt\_tab TYPE STANDARD TABLE OF ls\_tab,  
\*       wa\_tab TYPE ls\_tab.  
\*  
\*SELECT-OPTIONS s\_vbeln FOR wa\_tab-vbeln.  
\*  
\*START-OF-SELECTION.  
\*  
\*  SELECT so\_h~vbeln  
\*    so\_h~audat  
\*    so\_h~vbtyp  
\*    so\_h~trvog  
\*    so\_h~auart  
\*  
\*     so\_i~posnr  
\*     so\_i~matnr  
\*    so\_i~matwa  
\*  
\*    INTO TABLE lt\_tab  
\*    FROM vbak AS so\_h  
\*  
\*    INNER JOIN vbap AS so\_i  
\*    ON so\_h~vbeln = so\_i~vbeln  
\*  
\*    WHERE so\_h~vbeln IN s\_vbeln.  
\*  
\*  IF sy-subrc EQ 0.  
\*    WRITE : 'sales order info ', sy-dbcnt.  
\*  ELSE.  
\*    WRITE : 'sales order records are not found ', sy-dbcnt.  
\*  ENDIF.  
\*  
\*END-OF-SELECTION.  
\*  LOOP AT lt\_tab INTO wa\_tab.  
\*    WRITE : / wa\_tab-vbeln,  
\*     wa\_tab-posnr,  
\*    wa\_tab-matnr,  
\*    wa\_tab-audat,  
\*    wa\_tab-vbtyp,  
\*    wa\_tab-trvog,  
\*    wa\_tab-auart,  
\*       wa\_tab-matwa.  
\*  
\*  ENDLOOP.  
  
  
\*\*\*\*\*#  3  leftouter join  
  
  
  
  
\*\*\*\*# 2 two tables with alias  
\*TYPES: BEGIN OF ls\_tab,  
\*         vbeln TYPE vbeln\_va,  "  vbak  
\*         audat TYPE audat,  
\*         vbtyp TYPE vbtypl,  
\*         trvog TYPE trvog,  
\*         auart TYPE auart,  
\*  
\*         posnr TYPE posnr\_va,  "   vbap  
\*         matnr TYPE matnr,  
\*         matwa TYPE matwa,  
\*       END OF ls\_tab.  
\*  
\*DATA : lt\_tab TYPE STANDARD TABLE OF ls\_tab,  
\*       wa\_tab TYPE ls\_tab.  
\*  
\*SELECT-OPTIONS s\_vbeln FOR wa\_tab-vbeln.  
\*  
\*START-OF-SELECTION.  
\*  
\*  SELECT so\_h~vbeln  
\*    so\_h~audat  
\*    so\_h~vbtyp  
\*    so\_h~trvog  
\*    so\_h~auart  
\*  
\*     so\_i~posnr  
\*     so\_i~matnr  
\*    so\_i~matwa  
\*  
\*    INTO TABLE lt\_tab  
\*    FROM vbak AS so\_h  
\*  
\*     LEFT OUTER JOIN vbap AS so\_i  
\*    ON so\_h~vbeln = so\_i~vbeln  
\*  
\*    WHERE so\_h~vbeln IN s\_vbeln.  
\*  
\*  IF sy-subrc EQ 0.  
\*    WRITE : 'sales order info ', sy-dbcnt.  
\*  ELSE.  
\*    WRITE : 'sales order records are not found ', sy-dbcnt.  
\*  ENDIF.  
\*  
\*END-OF-SELECTION.  
\*  LOOP AT lt\_tab INTO wa\_tab.  
\*    WRITE : / wa\_tab-vbeln,  
\*     wa\_tab-posnr,  
\*    wa\_tab-matnr,  
\*    wa\_tab-audat,  
\*    wa\_tab-vbtyp,  
\*    wa\_tab-trvog,  
\*    wa\_tab-auart,  
\*       wa\_tab-matwa.  
\*  
\*  ENDLOOP.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_JOIN2  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_JOIN2.  
  
  
\*\*types: BEGIN OF ls\_tab,  
\*\*    MATNR TYPE MATNR,  "mara  
\*\*    ERNAM TYPE ERNAM,  
\*\*    LAEDA  TYPE LAEDA,  
\*\*    AENAM  TYPE AENAM,  
\*\*    VPSTA  TYPE VPSTA,  
\*\*  
\*\*    PSTAT TYPE PSTAT\_D,  "marc  
\*\*    LVORM  TYPE LVOWK,  
\*\*    BWTTY  TYPE BWTTY\_D,  
\*\*    XCHAR  TYPE XCHAR,  
\*\*  END OF LS\_TAB.  
\*\*  
\*\*  data: lt\_tab type STANDARD TABLE OF ls\_tab,  
\*\*        wa\_tab type ls\_tab,  
\*\*        lv\_mat type matnr.  
\*\*  
\*\*  select-OPTIONS sc\_mat for lv\_mat.  
\*\*  
\*\*  START-OF-SELECTION.  
\*\*  
\*\*  select  
\*\*    mara~matnr  
\*\*    mara~ernam  
\*\*    mara~laeda  
\*\*    mara~aenam  
\*\*    mara~vpsta  
\*\*    marc~pstat  
\*\*    marc~lvorm  
\*\*    marc~bwtty  
\*\*    marc~xchar  
\*\* into TABLE lt\_tab  
\*\*    from mara  
\*\*    left outer join marc  
\*\*    on mara~matnr = marc~matnr  
\*\*  
\*\*    where mara~matnr in sc\_mat.  
\*\*IF sy-subrc eq 0.  
\*\*  write: 'Records found and count is:', sy-dbcnt.  
\*\*  else.  
\*\*    write: 'Records not found count is:', sy-dbcnt.  
\*\*    ENDIF.  
\*\*  
\*\*end-of-SELECTION.  
\*\*  
\*\*loop at lt\_tab into wa\_tab.  
\*\*  write:/  
\*\*    wa\_tab-matnr,  
\*\*    wa\_tab-ernam,  
\*\*    wa\_tab-laeda,  
\*\*    wa\_tab-aenam,  
\*\*    wa\_tab-vpsta,  
\*\*    wa\_tab-pstat,  
\*\*    wa\_tab-lvorm,  
\*\*    wa\_tab-bwtty,  
\*\*    wa\_tab-xchar.  
\*\*  ENDLOOP.  
  
 TYPES: BEGIN OF ls\_tab,  
         matnr TYPE matnr,  "mara  
         ernam TYPE ernam,  
         laeda  TYPE laeda,  
         aenam  TYPE aenam,  
         vpsta  TYPE vpsta,  
  
         pstat TYPE pstat\_d,  "marc  
         lvorm  TYPE lvowk,  
         bwtty  TYPE bwtty\_d,  
         xchar  TYPE xchar,  
  
         spras TYPE spras,  "makt  
         maktx  TYPE maktx,  
         maktg  TYPE maktg,  
       END OF ls\_tab.  
  
DATA: lt\_tab TYPE STANDARD TABLE OF ls\_tab,  
      wa\_tab TYPE ls\_tab,  
      lv\_mat TYPE matnr.  
  
SELECT-OPTIONS sc\_mat FOR lv\_mat.  
  
START-OF-SELECTION.  
  
  SELECT  
    mara~matnr  
    mara~ernam  
    mara~laeda  
    mara~aenam  
    mara~vpsta  
    marc~pstat  
    marc~lvorm  
    marc~bwtty  
    marc~xchar  
    makt~spras  
    makt~maktx  
    makt~maktg  
 INTO TABLE lt\_tab  
    FROM mara  
    INNER JOIN marc  
    ON mara~matnr = marc~matnr  
    INNER JOIN makt  
    ON mara~matnr = makt~matnr  
  
    WHERE mara~matnr IN sc\_mat.  
  
END-OF-SELECTION.  
  
IF sy-subrc eq 0.  
  write: 'Records found and count is:', sy-dbcnt.  
  else.  
    write: 'Records not found count is:', sy-dbcnt.  
    ENDIF.  
  LOOP AT lt\_tab INTO wa\_tab.  
    WRITE:/  
      wa\_tab-matnr color 4,  
      10 wa\_tab-ernam COLOR 1,  
      20 wa\_tab-laeda COLOR 2,  
      35 wa\_tab-aenam COLOR 3,  
      45 wa\_tab-vpsta COLOR 4,  
      55 wa\_tab-pstat COLOR 5,  
      65 wa\_tab-lvorm COLOR 6,  
      75 wa\_tab-bwtty COLOR 7,  
      85 wa\_tab-xchar COLOR 1,  
      100 wa\_tab-spras COLOR 2,  
      140 wa\_tab-maktx COLOR 3,  
      160 wa\_tab-maktg COLOR 4.  
  ENDLOOP.

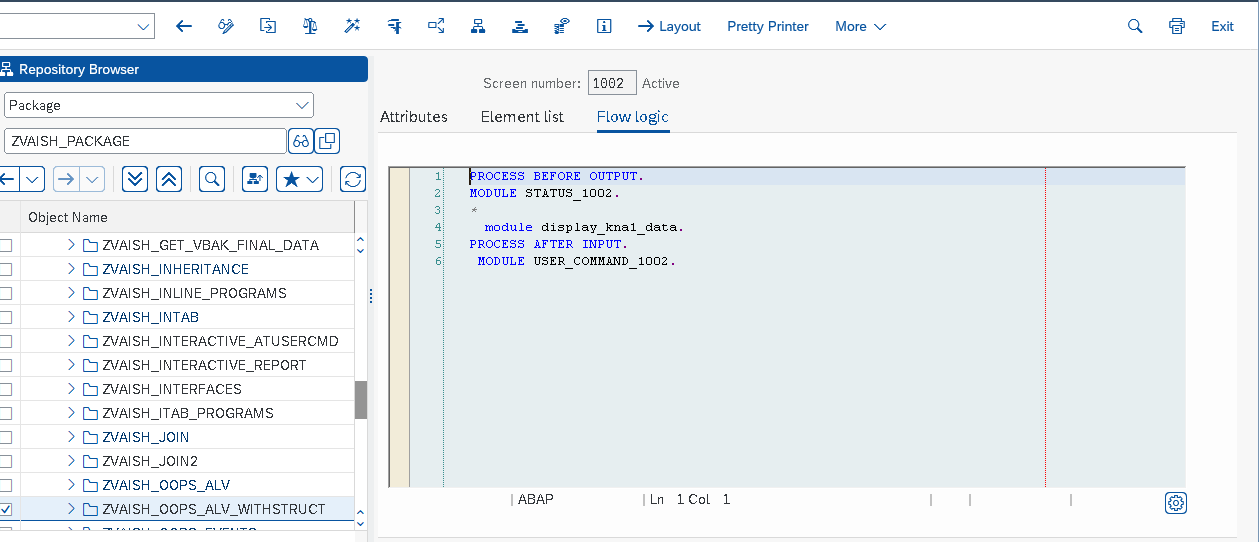
\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_OOPS\_ALV  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_OOPS\_ALV.  
  
data: lt\_tab type table of kna1, "internal table for collecting data  
      lv\_kunnr type kunnr. "parameter for selection.  
  
DATA: i\_container type ref to cl\_gui\_custom\_container,  "declaring conatiner object for craetion  
      i\_alvgrid type ref to cl\_gui\_alv\_grid. "declaring alv object for creating.  
  
select-options s\_kunnr for lv\_kunnr.  
START-OF-SELECTION.  
  
select \*  
  into table lt\_tab  
  from kna1  
   where kunnr in s\_kunnr.  
  
  call screen 1001.  
\*&---------------------------------------------------------------------\*  
\*& Module STATUS\_1001 OUTPUT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
MODULE status\_1001 OUTPUT.  
SET PF-STATUS 'ZVAISH\_PUSHBUTTONS'.  
\* SET TITLEBAR 'xxx'.  
ENDMODULE.  
\*&---------------------------------------------------------------------\*  
\*&      Module  USER\_COMMAND\_1001  INPUT  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
MODULE user\_command\_1001 INPUT.  
DATA ok\_code type SY-UCOMM.  
  
CASE ok\_code.  
  when 'BACK'.  
    LEAVE to SCREEN 0.  
    when 'EXIT'.  
      LEAVE  to SCREEN 0.  
  ENDCASE.  
  
ENDMODULE.  
\*&---------------------------------------------------------------------\*  
\*& Module OOALV\_INFO OUTPUT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
MODULE ooalv\_info OUTPUT.  
CREATE OBJECT i\_container  
  EXPORTING  
\*   parent                      =  
    container\_name              = 'CNT'  
\*    style                       =  
\*    lifetime                    = lifetime\_default  
\*    repid                       =  
\*    dynnr                       =  
\*    no\_autodef\_progid\_dynnr     =  
\*  EXCEPTIONS  
\*    cntl\_error                  = 1  
\*    cntl\_system\_error           = 2  
\*    create\_error                = 3  
\*    lifetime\_error              = 4  
\*    lifetime\_dynpro\_dynpro\_link = 5  
\*    others                      = 6  
    .  
IF sy-subrc <> 0.  
\* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO  
\*            WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.  
ENDIF.  
  
CREATE OBJECT i\_alvgrid  
  EXPORTING  
\*    i\_shellstyle      = 0  
\*    i\_lifetime        =  
    i\_parent          = i\_container  
\*    i\_appl\_events     = SPACE  
\*    i\_parentdbg       =  
\*    i\_applogparent    =  
\*    i\_graphicsparent  =  
\*    i\_name            =  
\*    i\_fcat\_complete   = SPACE  
\*    o\_previous\_sral\_handler =  
\*    i\_use\_one\_ux\_appearance = ABAP\_FALSE  
\*    o\_common\_hub      =  
\*    o\_function\_tool   =  
\*  EXCEPTIONS  
\*    error\_cntl\_create = 1  
\*    error\_cntl\_init   = 2  
\*    error\_cntl\_link   = 3  
\*    error\_dp\_create   = 4  
\*    others            = 5  
    .  
IF sy-subrc <> 0.  
\* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO  
\*            WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.  
ENDIF.  
  
CALL METHOD i\_alvgrid->set\_table\_for\_first\_display  
 EXPORTING  
\*    i\_buffer\_active               =  
\*    i\_bypassing\_buffer            =  
\*    i\_consistency\_check           =  
    i\_structure\_name              = 'KNA1'  
\*    is\_variant                    =  
\*    i\_save                        =  
\*    i\_default                     = 'X'  
\*    is\_layout                     =  
\*    is\_print                      =  
\*    it\_special\_groups             =  
\*    it\_toolbar\_excluding          =  
\*    it\_hyperlink                  =  
\*    it\_alv\_graphics               =  
\*    it\_except\_qinfo               =  
\*    ir\_salv\_adapter               =  
  CHANGING  
    it\_outtab                     = lt\_tab  
\*    it\_fieldcatalog               =  
\*    it\_sort                       =  
\*    it\_filter                     =  
\*  EXCEPTIONS  
\*    invalid\_parameter\_combination = 1  
\*    program\_error                 = 2  
\*    too\_many\_lines                = 3  
\*    others                        = 4  
        .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
  
ENDMODULE.

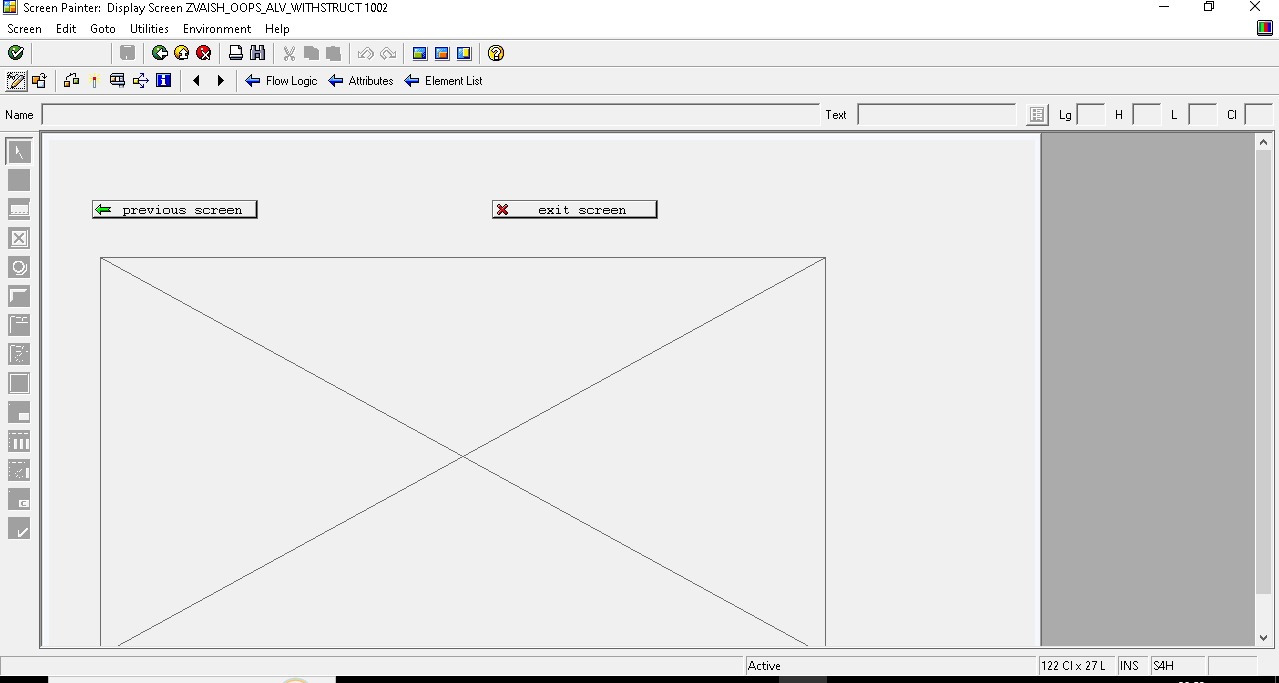
\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_OOPS\_ALV\_WITHSTRUCT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_OOPS\_ALV\_WITHSTRUCT.  
include zvaish\_kna1\_top.  
  
START-OF-SELECTION.  
  
perform get\_data.  
perform alv\_fcat.  
call screen 1002.  
  
INCLUDE zvaish\_oops\_alv\_getf01. "SQL QUERY TO GET DATA INTO LT\_TAB  
  
INCLUDE zvaish\_withstruct\_alvf01. "FIELDCATALOGUE TO PRINT SOME COLUMNS ONLY  
  
INCLUDE zvaish\_oops\_alv\_withstruct\_o01.

\*&---------------------------------------------------------------------\*  
\*& Include          ZVAISH\_KNA1\_TOP  
\*&---------------------------------------------------------------------\*  
  
types: BEGIN OF ls\_tab,  
  
  KUNNR   type KUNNR,  
LAND1  type LAND1\_GP,  
NAME1  type NAME1\_GP,  
NAME2  type NAME2\_GP,  
ORT01   type ORT01\_GP,  
  end of ls\_tab.  
  data : lt\_tab type STANDARD TABLE OF ls\_tab,  
        wa\_fieldcat type lvc\_s\_fcat, "fieldcats difened by sap for oops alv  
        lt\_fieldcat type lvc\_t\_fcat,  
        lv\_kunnr type kunnr,  
         i\_container type ref to cl\_gui\_custom\_container,  
       i\_alvgrid   type ref to cl\_gui\_alv\_grid,  
       LS\_layout TYPE lvc\_s\_layo.  
  
select-OPTIONS s\_kunnr for lv\_kunnr.

\*----------------------------------------------------------------------\*  
\*\*\*INCLUDE ZVAISH\_OOPS\_ALV\_GETF01.  
\*----------------------------------------------------------------------\*  
\*&---------------------------------------------------------------------\*  
\*& Form get\_data  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM get\_data .  
  
  SELECT KUNNR LAND1 NAME1 NAME2 ORT01  
    into table lt\_tab  
    from kna1  
    where kunnr in s\_kunnr.  
  
    if sy-subrc eq 0.  
\*      message (zvaish\_messages )  
      write:'records are found'.  
      else.  
        write:'records are not found'.  
        endif.  
  
  
  
ENDFORM.

\*----------------------------------------------------------------------\*  
\*\*\*INCLUDE ZVAISH\_WITHSTRUCT\_ALVF01.  
\*----------------------------------------------------------------------\*  
\*&---------------------------------------------------------------------\*  
\*& Form alv\_fcat  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM  
  alv\_fcat .  
  
  wa\_fieldcat-col\_pos = '1'.  
  wa\_fieldcat-fieldname = text-001.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'CUST NUM'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '2'.  
  wa\_fieldcat-fieldname = text-002.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'CUST COUNTRY'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '3'.  
  wa\_fieldcat-fieldname = text-003.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'FIRST NAME'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '4'.  
  wa\_fieldcat-fieldname = text-004.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'LAST NAME'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
  
  
  
  wa\_fieldcat-col\_pos = '5'.  
  wa\_fieldcat-fieldname = text-005.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'CUST CITY'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
ENDFORM.





\*----------------------------------------------------------------------\*  
\*\*\*INCLUDE ZVAISH\_OOPS\_ALV\_GETF01.  
\*----------------------------------------------------------------------\*  
\*&---------------------------------------------------------------------\*  
\*& Form get\_data  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM get\_data .  
  
  SELECT KUNNR LAND1 NAME1 NAME2 ORT01  
    into table lt\_tab  
    from kna1  
    where kunnr in s\_kunnr.  
  
    if sy-subrc eq 0.  
\*      message (zvaish\_messages )  
      write:'records are found'.  
      else.  
        write:'records are not found'.  
        endif.  
  
  
  
ENDFORM.

\*----------------------------------------------------------------------\*  
\*\*\*INCLUDE ZVAISH\_WITHSTRUCT\_ALVF01.  
\*----------------------------------------------------------------------\*  
\*&---------------------------------------------------------------------\*  
\*& Form alv\_fcat  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM  
  alv\_fcat .  
  
  wa\_fieldcat-col\_pos = '1'.  
  wa\_fieldcat-fieldname = text-001.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'CUST NUM'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '2'.  
  wa\_fieldcat-fieldname = text-002.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'CUST COUNTRY'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '3'.  
  wa\_fieldcat-fieldname = text-003.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'FIRST NAME'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
    wa\_fieldcat-col\_pos = '4'.  
  wa\_fieldcat-fieldname = text-004.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'LAST NAME'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
  
  
  
  wa\_fieldcat-col\_pos = '5'.  
  wa\_fieldcat-fieldname = text-005.  
  wa\_fieldcat-tabname = 'LT\_TAB'.  
  wa\_fieldcat-coltext = 'CUST CITY'.  
  append wa\_fieldcat to lt\_fieldcat.  
  clear wa\_fieldcat.  
  
ENDFORM.

\*----------------------------------------------------------------------\*  
\*\*\*INCLUDE ZVAISH\_OOPS\_ALV\_WITHSTRUCT\_O01.  
\*----------------------------------------------------------------------\*  
\*&---------------------------------------------------------------------\*  
\*& Module STATUS\_1002 OUTPUT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
MODULE status\_1002 OUTPUT.  
 SET PF-STATUS 'ZVAISH\_PUSHBUTTONS'.  
\* SET TITLEBAR 'xxx'.  
ENDMODULE.  
\*&---------------------------------------------------------------------\*  
\*&      Module  USER\_COMMAND\_1002  INPUT  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
MODULE user\_command\_1002 INPUT.  
  data ok\_code type SY-UCOMM.  
  
  CASE ok\_code.  
    when 'BACK'.  
      leave screen .  
      when 'CANCEL'.  
      leave to screen 0.  
    ENDCASE.  
  
ENDMODULE.  
\*&---------------------------------------------------------------------\*  
\*& Module DISPLAY\_KNA1\_DATA OUTPUT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
MODULE display\_kna1\_data OUTPUT.  
  
CREATE OBJECT i\_container  
  EXPORTING  
\*    parent                      =  
    container\_name              = 'CNT'  
\*    style                       =  
\*    lifetime                    = lifetime\_default  
\*    repid                       =  
\*    dynnr                       =  
\*    no\_autodef\_progid\_dynnr     =  
\*  EXCEPTIONS  
\*    cntl\_error                  = 1  
\*    cntl\_system\_error           = 2  
\*    create\_error                = 3  
\*    lifetime\_error              = 4  
\*    lifetime\_dynpro\_dynpro\_link = 5  
\*    others                      = 6  
    .  
IF sy-subrc <> 0.  
\* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO  
\*            WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.  
ENDIF.  
  
CREATE OBJECT i\_alvgrid  
  EXPORTING  
\*    i\_shellstyle      = 0  
\*    i\_lifetime        =  
    i\_parent          = i\_container  
\*    i\_appl\_events     = SPACE  
\*    i\_parentdbg       =  
\*    i\_applogparent    =  
\*    i\_graphicsparent  =  
\*    i\_name            =  
\*    i\_fcat\_complete   = SPACE  
\*    o\_previous\_sral\_handler =  
\*    i\_use\_one\_ux\_appearance = ABAP\_FALSE  
\*    o\_common\_hub      =  
\*    o\_function\_tool   =  
\*  EXCEPTIONS  
\*    error\_cntl\_create = 1  
\*    error\_cntl\_init   = 2  
\*    error\_cntl\_link   = 3  
\*    error\_dp\_create   = 4  
\*    others            = 5  
.  
  
IF sy-subrc <> 0.  
\* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO  
\*            WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.  
ENDIF.  
  
  
  
CALL METHOD i\_alvgrid->set\_table\_for\_first\_display  
 EXPORTING  
\*    i\_buffer\_active               =  
\*    i\_bypassing\_buffer            =  
\*    i\_consistency\_check           =  
   i\_structure\_name              = 'LS\_TAB'  
\*    is\_variant                    =  
\*    i\_save                        =  
\*    i\_default                     = 'X'  
   is\_layout                     = LS\_LAYOUT  
\*    is\_print                      =  
\*    it\_special\_groups             =  
\*    it\_toolbar\_excluding          =  
\*    it\_hyperlink                  =  
\*    it\_alv\_graphics               =  
\*    it\_except\_qinfo               =  
\*    ir\_salv\_adapter               =  
  CHANGING  
    it\_outtab                     = LT\_TAB  
    it\_fieldcatalog               = LT\_FIELDCAT  
\*    it\_sort                       =  
\*    it\_filter                     =  
\*  EXCEPTIONS  
\*    invalid\_parameter\_combination = 1  
\*    program\_error                 = 2  
\*    too\_many\_lines                = 3  
\*    others                        = 4  
        .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
ENDMODULE.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_OOPS\_EVENTS  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_OOPS\_EVENTS.  
  
  
\*\*\*\*\*\*\*\*oops events are specially to call methods of other class using events which needs to be trigggered  
  
\*#1.  
\*  
\*class c1 DEFINITION.  
\*  public section.  
\*  EVENTS E1. "e1 is event. we have created event here  
\*  METHODS M1.  
\*  METHODS T1 FOR  EVENT E1 of c1. "t1 is event handler method  
\*  endclass.  
\*  
\*  class c1 IMPLEMENTATION.  
\*    method m1.  
\*      write: /'i am normal method of class c1 AND I AM RAISING THE EVENT '.  
\*      RAISE EVENT E1.  
\*      ENDMETHOD.  
\*      method t1.  
\*        write: /'i am event handler method of class c1'.  
\*        ENDMETHOD.  
\*    endclass.  
\*  
\*    START-OF-SELECTION.  
\*  
\*    data obj type REF To c1.  
\*    create OBJECT obj.  
\*  
\*     SET HANDLER obj->T1 for obj. "first you need to register that event to get it recognised.  
\*  
\*  obj->m1( ). "then call the method in which event is there. otherwise event is not triggered .  
\*\*it doenot give error but event is not called.  
  
  
\*\*\*\*\*\*\*#2 event handler method in different class  
  
class c1 DEFINITION.  
  public section.  
  EVENTS E1. "e1 is event. we have created event here  
  METHODS M1.  
  
  endclass.  
  
  class c1 IMPLEMENTATION.  
    method m1.  
      write: /'i am normal method of class c1 AND I AM RAISING THE EVENT '.  
      RAISE EVENT E1.  
      ENDMETHOD.  
  
    endclass.  
  
class c2 definition.  
  public section.  
  methods: t1 FOR EVENT  E1 OF C1.  
  endclass.  
  
  class c2 IMPLEMENTATION.  
  
    method t1.  
      WRITE:/ 'I AM EVENT HANDLER METHOD IN CLASS 2'.  
  
      ENDMETHOD.  
    endclass.  
    START-OF-SELECTION.  
  
    data obj1 type REF To c1.  
  
    create OBJECT obj1.  
  
    DATA obj2 type REF to c2.  
    create  object obj2.  
  
    obj1->m1( ). "normally calling methods  
    obj2->t1( ).  
  
\*set handler obj2->t1 for obj2. "in c1 we are raising in c2 it is triggered  
\*t1 is only defined for triggereing objects of c1.  
set HANDLER obj2->t1 for obj1."registering the event handler method from c2 for c1  
obj2->t1( ).

\*\*&---------------------------------------------------------------------\*  
\*\*& Report ZVAISH\_OOPS\_METHODS  
\*\*&---------------------------------------------------------------------\*  
\*\*&  
\*\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_OOPS\_METHODS.  
\*  
\*\*                 #1 Use of STATIC Attributes  
  
\*Class c1 DEFINITION.  
\*  PUBLIC SECTION.  
\* class-data : num type I.  
\* methods : M1.  
\*  
\*  ENDCLASS.  
\*  
\*  CLASS C1 IMPLEMENTATION.  
\*    METHOD M1.  
\*      NUM = NUM + 1.  
\*      WRITE : 'before assing num value',NUM.  
\*      ENDMETHOD.  
\*    ENDCLASS.  
\*  
\*  
\* START-OF-SELECTION. "without start of selection object is not created  
\*  
\*    data: obj type REF TO C1.  
\*    CREATE OBJECT obj. "error comes in this line that statmt is not accessible  
\*     obj->m1( ).  
\*    c1=>num = 5.  
\*    write :'after assingingg',c1=>num.  
  
\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#2 different ways of calling the METHOD with one parameter.  
\*  
\*\*class c2 DEFINITION.  
\*\*  public SECTION.  
\*\*  data: v1 type I value '10'.  
\*\*  methods: M1 importing I\_input1 type I.  
\*\*  endclass.  
\*\*  
\*\*  class c2 IMPLEMENTATION.  
\*\*  
\*\*    METHOD M1.  
\*\*      v1 = v1 + I\_input1.  
\*\*      I\_input1 = 8. "cannot be modifies as it is reference type.  
\*\*      write v1.  
\*\*      ENDMETHOD.  
\*\*  
\*\*    ENDCLASS.  
\*\*  
\*\*    START-OF-SELECTION.  
\*\*    data: obj type ref to c2.  
\*\*    create object obj.  
\*\*\*    obj->M1( 10 ).  
\*\*\*    call method obj->M1 EXPORTING I\_input1 = '90'.  
\*\*\*    call method obj->M1( I\_input1 = 10 ).  
\*\*    call method obj->M1( 30 ).  
\*\*  
\*  
\*\*      #3 import parameters passed by ref.  
\*  
\*PARAMETERS p\_num type i.  
\*  
\*  
\*class C1 definition.  
\*  public SECTION.  
\*  methods M1 IMPORTING I\_input1 type I  VALUE(input2) type I.  
\*  
\*  
\*  
\*  ENDCLASS.  
\*  
\*  class c1 IMPLEMENTATION.  
\*    METHOD M1.  
\*    INPUT2 = 7.  
\*i\_input1 = 14. "cannot be modified.  
\*      WRITE : 'input 2 value is', input2.  
\*  
\*      ENDMETHOD.  
\*  
\*    ENDCLASS.  
\*    START-OF-SELECTION.  
\*    data obj type REF TO c1.  
\*    create object obj.  
\*  
\*call method obj->M1 exporting I\_input1 = 10 input2 = p\_num.  
\*  
\*  
\*\* #4 use of EXPORT AND CHANGING PARAMETERS OF A METHOD  
  
\*PARAMETERS p\_grade(10) type c.  
\*  
\*data: lv\_tax type p decimals 2,  
\*      lv\_salary type p decimals 2.  
\*  
\*class ctax definition.  
\*  PUBLIC SECTION.  
\*  METHODS : tax\_calc importing grade type c  
\*            exporting itax type p  
\*       salary type p.  
\*  ENDCLASS.  
\*  
\*  class ctax IMPLEMENTATION.  
\*    METHOD tax\_calc.  
\*      case grade.  
\*        when 'A01'.  
\*          itax = salary \* '0.2'.  
\*          when 'A02'.  
\*            itax = salary \* '0.1'.  
\*            when others.  
\*              itax = salary \* '0.15'.  
\*              endcase.  
\*              salary = salary - itax.  
\*                  endmethod.  
\*    ENDCLASS.  
\*  
\*    START-OF-SELECTION.  
\*    data obj type REF TO c1.  
\*    create object obj.  
\*  
\*    lv\_salary = 10000.  
\*    lv\_tax = 0.  
\*    write : 'before calling method, salary cand tax values are' , lv\_salary, lv\_tax.  
\*    call method obj->tax\_calc  
\*    exporting grade = 'A01'  
\*      importing itax = lv\_tax  
\*        changing salary = lv\_salary.  
\*    write : 'after calling method the values of salary and tax are',lv\_salary,lv\_tax.  
  
  
\*\*\* #5 use of returning parameters in method  
\*  
\*\*class c1 DEFINITION.  
\*\*  public SECTION.  
\*\*  methods m1 IMPORTING num1 type i  
\*\*    exporting num2 type i  
\*\*      RETURNING VALUE(result) type I  
\*\*      EXCEPTIONS e1.  
\*\*  ENDCLASS.  
\*\*  
\*\*  class c1 IMPLEMENTATION.  
\*\*    method m1.  
\*\*      if num1 LT 20.  
\*\*        message i007(zvaish\_messages) with 'value should be more than 20' raising e1.  
\*\*      else.  
\*\*        num2 = num1 + 10.  
\*\*      result = num2.  
\*\*      ENDIF.  
\*\*      ENDMETHOD.  
\*\*    ENDCLASS.  
\*\*    START-OF-SELECTION.  
\*\*    data w\_num type i.  
\*\*  
\*\*    data obj type REF TO c1.  
\*\*    create object obj.  
\*\*  
\*\*\*    obj->m1( 10 )  "is it only for just sending values.  
\*\*    call method : obj->m1  
\*\*    exporting num1 = 100  
\*\*    receiving result = w\_num.  
\*\*  
\*\*  
\*\*        write w\_num.  
\*  
\*  
\*  
\*\*  
\*\*\*\*\*\*\*\*\*\*\*\*#6 STATIC METHODS can on;y use static attributes  
\*  
\*   class c1 DEFINITION.  
\*     public section.  
\*     class-data: stnum type i value 10.  
\*     data: insnum type i value 20.  
\*  
\*     class-methods: stmeth.  
\*     methods: instmeth.  
\*  
\*     endclass.  
\*  
\*     class c1 IMPLEMENTATION.  
\*       method stmeth.  
\*         write: stnum.  
\*\*write: insnum.  
\*         ENDMETHOD.  
\*  
\*         method instmeth.  
\*           write : stnum.  
\*\*write: insnum. "instance varibale cannot be accessed in static method.  
\*           ENDMETHOD.  
\*  
\*       endclass.  
\*       START-OF-SELECTION.  
\*       data obj type REF TO c1.  
\*         call method c1=>stmeth.  
\*       create object obj.  
\*  
\*       call method obj->instmeth.  
\*  
\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*#7 EXCEPTION RAISING\*\*\*\*\*\*\*\*\*\*  
\*  
\*\*data: w\_num type I.  
\*\*class c1 definition.  
\*\*  public section.  
\*\*  methods  M1 importing num1 type I  
\*\*              exporting num2 type I  
\*\*         returning VALUE(result) type I  
\*\*      exceptions e1.  
\*\*  
\*\*  endclass.  
\*\*  
\*\*  class c1 IMPLEMENTATION .  
\*\*    method m1.  
\*\*      if num1 LT 10.  
\*\*        message w007(zvaish\_messages) with 'num shoulb be more than 10' raising e1.     "in message class 007 has & that is we can dynamically use that field and insert our own messages suchnas exception .  
\*\*  
\*\*      else.  
\*\*        num2 = num1 \* 5.  
\*\*result = num2.  
\*\*        ENDIF.  
\*\*      ENDMETHOD.  
\*\*    ENDCLASS.  
\*\*  
\*\*    PARAMETERS: p\_no type I.  
\*\*    START-OF-SELECTION.  
\*\*  
\*\*    data obj type ref to c1.  
\*\*    create object obj.  
\*\*    obj->M1( p\_no ).  
\*\*    obj->m1( p\_no ).  
\*\*   call method obj->m1 exporting num1 = p\_no  
\*\*  
\*\*                     receiving result = w\_num.  
\*\*  
\*\*    if sy-subrc = 4.  
\*\*      write 'unable to calculate'.  
\*\*      else.  
\*\*        write w\_num.  
\*\*        ENDIF.  
\*  
\*  
\*\*\*\*\*\*\*\*\*\*\*#8. constructor  
\*data num type i value '12'.  
\*class c1 DEFINITION.  
\*write 'before visibility defining methods'. "this will not print gives run time  error as first the visibility should be there  
\*public section.  
\*write 'after visibility'.  
\*data num type i value '128'.  
\*class-Data  num2 type  i value 123.  
\*write num.             "we can just define not display.  
\*PROTECTED SECTION.         "always for constructor visisbility should be public  
\*  methods:m1,  
\*  constructor.  
\*  endclass.  
\*  
\*  class c1 IMPLEMENTATION.  
\*write :'inside implementation outside methods'. "only methods can be impleemneted here  
\*    method constructor.  
\*      write 'i am constructor method'.  
\*      write me->num.  
\*      endmethod.  
\*      method m1.  
\*        write 'i am instance method'.  
\*\*\*\*\*\*\*\*        write: obj->num.  
\*        endmethod.  
\*write num.         "only methods are implemented.  
\*    endclass.  
\*  
\*    START-OF-SELECTION.  
\*write num.            " to acces class atributes we need object  
\*    data obj TYPE REF TO c1.  
\*    create object obj.  
\*obj->m1( ).  
\*obj.me->num.  
\*write obj->num.  
\*write c1=>num2.  
\*\*\*\*\*\*\*obj->num.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*# 8.2 only instance constructor can have import parameters.  
\*\*  
\*class c1 DEFINITION.  
\*  public SECTION.  
\*  methods : constructor importing  I\_today type d. "constructor cannot have exporting parameter.  
\*  
\*  ENDCLASS.  
\*  
\*  class c1 IMPLEMENTATION.  
\*    method constructor.  
\*  
\*      write:/ 'today is :', I\_today .  
\*      endmethod.  
\*  
\*    endclass.  
\*    START-OF-SELECTION.  
\*    data obj type REF TO c1.  
\*    create object obj EXPORTING I\_today = sy-datum.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*8.3 Instance Constructors can raaise exceptions.  
\*PARAMETERS p\_num type i.  
\*class c1 DEFINITION.  
\*  public section.  
\*  class-data num1 type i value 100.  
\*  methods constructor importing num2 type I  
\*    exceptions e1.  
\*  
\*  endclass.  
\*  
\*  class c1 IMPLEMENTATION.  
\*    method constructor.  
\*      write 'i am instance constructor'.  
\*      if num2 LT 20.  
\*        RAISE e1.  
\*        else.  
\*      write :'num2 value passed to me is',num2.  
\*      endif.  
\*  
\*      endmethod.  
\*    endclass.  
\*  
\*    START-OF-SELECTION.  
\*    data obj type REF TO c1.  
\*    create object obj exporting num2 = p\_num  
\*          exceptions e1 = 4. "takes index value i.e sysubrc vlaue  
\*     if sy-subrc = 4.  
\*       write: 'number is less than 20'.  
\*       endif.  
\*    write c1=>num1.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*#8.4  static constructor can be triggered at the biggining of the processing block/form/event.  
\*class c1 DEFINITION.  
\*  public section.  
\*  class-data: num type i value  10.  
\*  methods constructor. "constructor is predefined for instance constructor.  
\*  class-methods class\_constructor. "class\_constructor is a predefined name otherwise acts like normal method but not as constructor.  
\*  ENDCLASS.  
\*  class c1 IMPLEMENTATION.  
\*    method constructor.  
\*      write 'im instance constructor'.  
\*      ENDMETHOD.  
\*      method class\_constructor.  
\*        write 'im static constructor'.  
\*        endmethod.  
\*    endclass.  
\*    START-OF-SELECTION.  
\*    data obj type REF TO c1.  
\*    create object obj.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* #9 ME keyword\*\*\*\*\*\*\*\*\*\*\*\*  
class c1 DEFINITION.  
  public section.  
\*data name type i value  10.  
data name type string value  'publicname'.    "not accepting char?  
  methods m1.  
  endclass.  
  
  class c1 IMPLEMENTATION.  
    method m1.  
data :name type i value 20.  
\*data name(10) type char value  20.  
      write name.  
      write me->name.  
      endmethod.  
    endclass.  
     START-OF-SELECTION.  
    data obj type REF TO c1.  
    create object obj.  
  
    obj->m1( ).  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*#10 iterative method(method calling itself)\*\*\*\*\*\*\*\*\*  
  
\*class c1 DEFINITION.  
\*  public section.  
\*  data var type i value 100.  
\*  methods m1.  
\*  endclass.  
\*  
\*  class c1 IMPLEMENTATION.  
\*    method m1.  
\*      write: 'i am instance method'.  
\*var = var + 5.  
\*  
\*  
\*  
\*      endmethod.  
\*    endclass.  
\*    START-OF-SELECTION.  
\*  
\*    data obj type REF TO c1.  
\*    create object obj.  
\*\*    do 5 times. "do stmts giving dump error  
\*\*    obj->m1.  
\*\*    ENDDO.  
\*    obj->m1( ). "if m1() no need to write call method  
\*   write obj->var.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_OREPORT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_OREPORT.  
  
include zvaish\_apreport.  
select-options  s\_aufnr for wa\_tab-aufnr.  
load-OF-PROGRAM.  
  
initialization.  
clear : lt\_tab , wa\_tab.  
s\_aufnr-low = '9212741323'.  
s\_aufnr-high = '9912741323'.  
append s\_aufnr.  
  
  
  
at SELECTION-SCREEN on s\_aufnr.  
  
  select single aufnr  
      into lv\_aufnr  
      from zvaish\_aufk  
      where aufnr in s\_aufnr.  
  
      if sy-subrc eq 0.  
        MESSAGE s000(zvaish\_messages).  
        else.  
          message e001(zvaish\_messages).  
        ENDIF.  
\*  START-OF-SELECTION.  
\*  top-OF-PAGE.  
\*  end-of-page.  
\*  end-of-SELECTION.  
        START-OF-SELECTION.  
 PERFORM get\_aufnr\_data.  
  
  
  
  PERFORM disp\_aufnr\_data.  
  
  TOP-OF-PAGE.  
  uline.  
  write:/  
  
  'Sales Document',  
10 'Record Created On',  
20 'Entry time',  
30 'Name of Person Responsible for Creating the Object',  
40 'Quotation/Inquiry is Valid From',  
50 'Sold-to Party'.  
  
  uline.  
  
 END-OF-PAGE.  
  write:/ 'date', sy-datum,  
  20 'time', sy-uzeit.  
  END-OF-SELECTION.  
  
  
  
  
  AT LINE-SELECTION.  
    CASE sy-lsind.  
      WHEN 1.  
        select AUFNR POSNR QUNUM QUPOS  
          into  table lt\_tab2  
          from zvaish\_afpo  
          where aufnr eq wa\_tab-aufnr.  
  
          loop at lt\_tab2 into wa\_tab2.  
            write :/ wa\_tab2-aufnr,  
            wa\_tab2-posnr,  
            wa\_tab2-qunum,  
            wa\_tab2-qupos.  
  
            ENDLOOP.  
  
  
    ENDCASE.  
  
  
  
  
  
FORM get\_aufnr\_data .  
  
  
  
  select AUFNR AUART AUTYP WERKS  
    into table lt\_tab  
    from zvaish\_aufk  
    where aufnr between s\_aufnr-low and s\_aufnr-high.  
  
  
ENDFORM.  
  
  
  
FORM disp\_aufnr\_data .  
  
  
  
  loop at lt\_tab into wa\_tab.  
        write :/ wa\_tab-aufnr,  
wa\_tab-auart,  
wa\_tab-autyp,  
wa\_tab-werks.  
        HIDE wa\_tab-aufnr.  
  
  
  
    ENDLOOP.  
  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_PRG  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_PRG.  
  
WRITE /: 'A'.  
  
WRITE /: 'B'.  
  
ENHANCEMENT-POINT ZVAISH\_ENHANCEMENT SPOTS ZVAISH\_ENHANCESPOT .  
\*$\*$-Start: ZVAISH\_ENHANCEMENT------------------------------------------------------------------$\*$\*  
ENHANCEMENT 1  ZVAISH\_ENHANCE\_IMPLEMENT.    "active version  
 write / : 'IMPLEMENTATION OF ENHANCEMENT'.  
ENDENHANCEMENT.  
\*$\*$-End:   ZVAISH\_ENHANCEMENT------------------------------------------------------------------$\*$\*  
  
WRITE /: 'C'.  
  
WRITE /: 'D'.  
WRITE /: 'E'.

\*&---------------------------------------------------------------------\*  
\*& Report zvaish\_procedure  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zvaish\_procedure.  
  
\* TYPES:BEGIN OF LS\_TAB,  
\*       VBELN TYPE VBELN\_VA,  
\*       ERNAM TYPE ERNAM,  
\*       ERZET TYPE ERZET,  
\*       VBTYP TYPE VBTYP,  
\*       END OF LS\_TAB.  
\*  
\*DATA: LT\_TAB TYPE STANDARD TABLE OF LS\_TAB WITH EMPTY KEY,  
\*      WA\_TAB TYPE LS\_TAB.  
\*  
\*SELECT-OPTIONS S\_VBELN FOR WA\_TAB-VBELN.  
\*START-OF-SELECTION.  
\*DATA OBJ TYPE REF TO zcl\_vaish\_class.  
\*CREATE OBJECT OBJ.  
\*OBJ->meth\_sales(  
\*  EXPORTING  
\*    i\_vbeln1 = S\_VBELN-LOW  
\*    i\_vbeln2 = S\_VBELN-HIGH  
\*  IMPORTING  
\*   lt\_tab   = LT\_TAB  
\*).  
\*LOOP AT LT\_TAB INTO WA\_TAB.  
\*WRITE: WA\_TAB-VBELN,  
\*WA\_TAB-erzet.  
\*  
\*ENDLOOP.  
  
  
  
\*  
\* TYPES:BEGIN OF LS\_TAB,  
\*       VBELN TYPE VBELN\_VA,  
\*       ERNAM TYPE ERNAM,  
\*       ERZET TYPE ERZET,  
\*       VBTYP TYPE VBTYP,  
\*       END OF LS\_TAB.  
\*  
\*DATA: LT\_TAB TYPE STANDARD TABLE OF LS\_TAB WITH EMPTY KEY,  
\*      WA\_TAB TYPE LS\_TAB.  
\*  
\*SELECT-OPTIONS S\_VBELN FOR WA\_TAB-VBELN.  
\*START-OF-SELECTION.  
\*DATA OBJ TYPE REF TO zcl\_vaish\_class.  
\*CREATE OBJECT OBJ.  
\*OBJ->meth\_sales(  
\*  EXPORTING  
\*    i\_vbeln1 = S\_VBELN-LOW  
\*    i\_vbeln2 = S\_VBELN-HIGH  
\*  IMPORTING  
\*   lt\_tab   = LT\_TAB  
\*).  
\*CL\_DEMO\_OUTPUT=>display\_data(  
\*  value   = LT\_TAB  
\*\*  name    =  
\*\*  exclude =  
\*\*  include =  
\*).  
  
  
\* TYPES:BEGIN OF LS\_TAB,  
\*       VBELN TYPE VBELN\_VA,  
\*       ERNAM TYPE ERNAM,  
\*       ERZET TYPE ERZET,  
\*       VBTYP TYPE VBTYP,  
\*       END OF LS\_TAB.  
\*  
\*DATA: LT\_TAB TYPE STANDARD TABLE OF LS\_TAB WITH EMPTY KEY,  
\*      WA\_TAB TYPE LS\_TAB.  
DATA LV\_VBELN TYPE VBELN\_VA.  
SELECT-OPTIONS S\_VBELN FOR LV\_VBELN.  
START-OF-SELECTION.  
DATA(OBJ) = NEW ZCL\_VAISH\_CLASS( ).  
OBJ->meth\_sales(  
  EXPORTING  
    i\_vbeln1 = S\_VBELN-LOW  
    i\_vbeln2 = S\_VBELN-HIGH  
  IMPORTING  
   lt\_tab   = DATA(LT\_TAB)  
).  
CL\_DEMO\_OUTPUT=>display\_data(  
  value   = LT\_TAB  
\*  name    =  
\*  exclude =  
\*  include =  
).

\*&---------------------------------------------------------------------\*  
\*& Report zvaish\_procejoin  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zvaish\_procejoin.  
DATA LV\_VBELN TYPE VBELN\_VA.  
SELECT-OPTIONS S\_VBELN FOR LV\_VBELN.  
START-OF-SELECTION .  
DATA OBJ TYPE REF TO ZVAISH\_PROCLS.  
CREATE OBJECT OBJ.  
OBJ->meth\_vbakap(  
  EXPORTING  
    i\_vbeln1 = s\_vbeln-low  
    i\_vbeln2 = s\_vbeln-high  
 IMPORTING  
    lt\_tab   = DATA(LT\_TAB)  
).  
  
CL\_DEMO\_OUTPUT=>display\_data(  
  value   = LT\_TAB  
\*  name    =  
\*  exclude =  
\*  include =  
).

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SELECTIONSCREEN  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_SELECTIONSCREEN.  
  
  
  
  
types : begin of ls\_tab,  " strs  
          matnr type matnr, " Material Number  
          mbrsh type mbrsh, " Industry  
          mtart type mtart, " Material Number  
          meins type meins, " Base Unit of Measure  
          brgew type brgew, " Gross Weight  
          ntgew type ntgew, " Net Weight  
  
        end of ls\_tab.  
  
data : lt\_tab type standard table of ls\_tab,  " itab  
       wa\_tab type ls\_tab.  " work area  
data: lv\_matnr type matnr.  
  
  
load-OF-PROGRAM.  
  
  
SELECTION-SCREEN : BEGIN OF BLOCK B1 WITH FRAME TITLE TEXT-001.  
  
\*  SELECT-OPTIONS S\_MATNR for lv\_matnr.  
SELECT-OPTIONS S\_MATNR for lv\_matnr OBLIGATORY NO INTERVALS.  
  
  SELECTION-SCREEN : END OF BLOCK B1.  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B2 WITH FRAME TITLE TEXT-002.  
     PARAMETERS p\_mtart type mtart.  
  
     SELECTION-SCREEN : END OF BLOCK B2.  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B3 WITH FRAME TITLE TEXT-003.  
    PARAMETERS : R1 RADIOBUTTON GROUP G1,  
    R2 RADIOBUTTON GROUP G1.  
  
  
  
    SELECTION-SCREEN  : ULINE /30(50).  
    SELECTION-SCREEN : COMMENT /50(20) lv\_name.  
    SELECTION-SCREEN: SKIP 2.  
    SELECTION-SCREEN: ULINE.  
  
  
  
  
  
    SELECTION-SCREEN : END OF BLOCK B3.  
    initialization.  
  
lv\_name  = 'UST HYD'.  
  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B4 WITH FRAME TITLE TEXT-004.  
  
  PARAMETERS:  c1 AS CHECKBOX DEFAULT 'X',  
    C2 AS CHECKBOX.  
  
    SELECTION-SCREEN : END OF BLOCK B4.  
  
  
  
  
  
  
  
START-OF-SELECTION.  
end-of-selection.  
  
  
  
  
start-of-selection.  
  perform get\_mat\_data.  
  perform disp\_mat\_data.  
  
\*&---------------------------------------------------------------------\*  
\*& Form get\_mat\_data  
\*&---------------------------------------------------------------------\*  
form get\_mat\_data .  
  
  select matnr mbrsh mtart meins brgew ntgew  
      into table lt\_Tab  
      from mara  
      where matnr in  s\_matnr.  
  if sy-subrc eq 0.  
    write :/ 'Records are found in mara table'.  
  else.  
    write : / 'Records are not found in mara table'.  
  endif.  
  
endform.  
\*&---------------------------------------------------------------------\*  
\*& Form disp\_mat\_data  
\*&---------------------------------------------------------------------\*  
  
form disp\_mat\_data .  
  loop at lt\_tab into wa\_tab.  
    write : / wa\_tab-matnr,  
              wa\_tab-mbrsh,  
              wa\_tab-mtart,  
              wa\_tab-meins,  
              wa\_tab-brgew,  
              wa\_tab-ntgew.  
  endloop.  
endform.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SELECTIONSCREEN\_OUTPUT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_SELECTIONSCREEN\_OUTPUT.  
  
  
  
  
types : begin of ls\_tab,  " strs  
          matnr type matnr, " Material Number  
          mbrsh type mbrsh, " Industry  
          mtart type mtart, " Material Number  
          meins type meins, " Base Unit of Measure  
          brgew type brgew, " Gross Weight  
          ntgew type ntgew, " Net Weight  
  
        end of ls\_tab.  
  
data : lt\_tab type standard table of ls\_tab,  " itab  
       wa\_tab type ls\_tab.  " work area  
data: lv\_matnr type matnr.  
  
  
load-OF-PROGRAM.  
  
  
SELECTION-SCREEN : BEGIN OF BLOCK B1 WITH FRAME TITLE TEXT-001.  
  
\*  SELECT-OPTIONS S\_MATNR for lv\_matnr.  
SELECT-OPTIONS S\_MATNR for lv\_matnr OBLIGATORY NO INTERVALS.  
  
  SELECTION-SCREEN : END OF BLOCK B1.  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B2 WITH FRAME TITLE TEXT-002.  
     PARAMETERS :p\_mtart type mtart MODIF ID M1,  
     P\_MBRSH type MBRSH MODIF ID M2.  
  
  
     SELECTION-SCREEN : END OF BLOCK B2.  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B3 WITH FRAME TITLE TEXT-003.  
    PARAMETERS : R1 RADIOBUTTON GROUP G1 USER-COMMAND FLAG DEFAULT 'X',  
    R2 RADIOBUTTON GROUP G1.  
  
  
  
    SELECTION-SCREEN  : ULINE /30(50).  
    SELECTION-SCREEN : COMMENT /50(20) lv\_name.  
    SELECTION-SCREEN: SKIP 2.  
    SELECTION-SCREEN: ULINE.  
  
  
  
  
  
    SELECTION-SCREEN : END OF BLOCK B3.  
    initialization.  
  
lv\_name  = 'UST HYD'.  
  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B4 WITH FRAME TITLE TEXT-004.  
  
  PARAMETERS:  c1 AS CHECKBOX DEFAULT 'X',  
    C2 AS CHECKBOX.  
  
    SELECTION-SCREEN : END OF BLOCK B4.  
  
    AT SELECTION-SCREEN OUTPUT.  
      LOOP AT SCREEN.  
        IF R1 = 'X' AND  
          SCREEN-GROUP1 = 'M2'.  
          SCREEN-ACTIVE = 0.  
  
          ELSEIF R2 = 'X' AND  
            SCREEN-GROUP1 = 'M1'.  
            SCREEN-ACTIVE = 0.  
            ENDIF.  
            MODIFY SCREEN.  
        ENDLOOP.  
  
  
  
  
  
  
  
START-OF-SELECTION.  
end-of-selection.  
  
  
  
  
start-of-selection.  
  perform get\_mat\_data.  
  perform disp\_mat\_data.  
  
\*&---------------------------------------------------------------------\*  
\*& Form get\_mat\_data  
\*&---------------------------------------------------------------------\*  
form get\_mat\_data .  
  
  select matnr mbrsh mtart meins brgew ntgew  
      into table lt\_Tab  
      from mara  
      where matnr in  s\_matnr.  
  if sy-subrc eq 0.  
    write :/ 'Records are found in mara table'.  
  else.  
    write : / 'Records are not found in mara table'.  
  endif.  
  
endform.  
\*&---------------------------------------------------------------------\*  
\*& Form disp\_mat\_data  
\*&---------------------------------------------------------------------\*  
  
form disp\_mat\_data .  
  loop at lt\_tab into wa\_tab.  
    write : / wa\_tab-matnr,  
              wa\_tab-mbrsh,  
              wa\_tab-mtart,  
              wa\_tab-meins,  
              wa\_tab-brgew,  
              wa\_tab-ntgew.  
  endloop.  
endform.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SELECT\_SCREEN\_REPORT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_SELECT\_SCREEN\_REPORT.  
  
  
types : begin of ls\_tab,  " strs  
          matnr type matnr, " Material Number  
          mbrsh type mbrsh, " Industry  
          mtart type mtart, " Material Number  
          meins type meins, " Base Unit of Measure  
          brgew type brgew, " Gross Weight  
          ntgew type ntgew, " Net Weight  
  
        end of ls\_tab.  
  
data : lt\_tab type standard table of ls\_tab,  " itab  
       wa\_tab type ls\_tab.  " work area  
data: lv\_matnr type matnr.  
  
  
load-OF-PROGRAM.  
  
  
SELECTION-SCREEN : BEGIN OF BLOCK B1 WITH FRAME TITLE TEXT-001.  
  
\*  SELECT-OPTIONS S\_MATNR for lv\_matnr.  
SELECT-OPTIONS S\_MATNR for lv\_matnr.  
  
  SELECTION-SCREEN : END OF BLOCK B1.  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B2 WITH FRAME TITLE TEXT-002.  
     PARAMETERS p\_mtart type mtart.  
  
     SELECTION-SCREEN : END OF BLOCK B2.  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B3 WITH FRAME TITLE TEXT-003.  
    PARAMETERS : R1 RADIOBUTTON GROUP G1,  
    R2 RADIOBUTTON GROUP G1.  
  
  
  
    SELECTION-SCREEN  : ULINE /30(50).  
    SELECTION-SCREEN : COMMENT /50(20) lv\_name.  
    SELECTION-SCREEN: SKIP 2.  
    SELECTION-SCREEN: ULINE.  
  
  
  
  
  
    SELECTION-SCREEN : END OF BLOCK B3.  
    initialization.  
  
lv\_name  = 'UST HYD'.  
  
  
  SELECTION-SCREEN : BEGIN OF BLOCK B4 WITH FRAME TITLE TEXT-004.  
  
  PARAMETERS:  c1 AS CHECKBOX DEFAULT 'X',  
    C2 AS CHECKBOX.  
  
    SELECTION-SCREEN : END OF BLOCK B4.  
  
  
  
  
  
  
  
START-OF-SELECTION.  
end-of-selection.  
  
  
  
  
start-of-selection.  
  perform get\_mat\_data.  
  perform disp\_mat\_data.  
  
\*&---------------------------------------------------------------------\*  
\*& Form get\_mat\_data  
\*&---------------------------------------------------------------------\*  
form get\_mat\_data .  
  
  select matnr mbrsh mtart meins brgew ntgew  
      into table lt\_Tab  
      from mara  
      where matnr in  s\_matnr.  
  if sy-subrc eq 0.  
    write :/ 'Records are found in mara table'.  
  else.  
    write : / 'Records are not found in mara table'.  
  endif.  
  
endform.  
\*&---------------------------------------------------------------------\*  
\*& Form disp\_mat\_data  
\*&---------------------------------------------------------------------\*  
  
form disp\_mat\_data .  
  loop at lt\_tab into wa\_tab.  
    write : / wa\_tab-matnr,  
              wa\_tab-mbrsh,  
              wa\_tab-mtart,  
              wa\_tab-meins,  
              wa\_tab-brgew,  
              wa\_tab-ntgew.  
  endloop.  
endform.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SFADOBE1\_PROGRAM  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_SFADOBE1\_PROGRAM.  
include zvaish\_sfadobe1\_top.  
  
  
PARAMETERS p\_vbeln type VBELN\_VA.  
\*SELECT-OPTIONS S\_VBELN FOR WA\_TAB-VBELN.  
  
START-OF-SELECTION.  
  
WA\_OUTPARAM-DEVICE = 'PRINTER'.  
WA\_OUTPARAM-dest = 'LP01'.  
WA\_OUTPARAM-NODIALOG = 'X'.  
WA\_OUTPARAM-PREVIEW = 'X'.  
  
  
CALL FUNCTION 'FP\_JOB\_OPEN'  
  CHANGING  
    ie\_outputparams       = WA\_OUTPARAM  
\* EXCEPTIONS  
\*   CANCEL                = 1  
\*   USAGE\_ERROR           = 2  
\*   SYSTEM\_ERROR          = 3  
\*   INTERNAL\_ERROR        = 4  
\*   OTHERS                = 5  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
START-OF-SELECTION.  
  
SELECT  VBELN POSNR MATNR MATWA INTO TABLE LT\_TAB FROM VBAP WHERE VBELN eq  p\_VBELN.  
  SELECT  SINGLE VBELN ERDAT ERZET KUNNR INTO WA\_TAB FROM VBAK WHERE VBELN eq p\_VBELN.  
  
    CALL FUNCTION 'FP\_FUNCTION\_MODULE\_NAME'  
      EXPORTING  
        i\_name                     = 'ZVAISH\_ADOBEFORM\_SFADOBE1'  
    IMPORTING  
       E\_FUNCNAME                 = GV\_FMNAME  
\*       E\_INTERFACE\_TYPE           =  
\*       EV\_FUNCNAME\_INBOUND        =  
              .  
  
CALL FUNCTION     GV\_FMNAME "'/1BCDWB/SM00000517'  
  EXPORTING  
  /1BCDWB/DOCPARAMS        = WA\_DOCPARAM  
    wa\_tab                   = WA\_TAB  
    lt\_tab                   = LT\_TAB  
\* IMPORTING  
\*   /1BCDWB/FORMOUTPUT       =  
\* EXCEPTIONS  
\*   USAGE\_ERROR              = 1  
\*   SYSTEM\_ERROR             = 2  
\*   INTERNAL\_ERROR           = 3  
\*   OTHERS                   = 4  
          .  
IF sy-subrc <> 0.  
\* Implement suitable error handling here  
ENDIF.  
  
  
  
  
  
    CALL FUNCTION 'FP\_JOB\_CLOSE'  
\*     IMPORTING  
\*       E\_RESULT             =  
\*     EXCEPTIONS  
\*       USAGE\_ERROR          = 1  
\*       SYSTEM\_ERROR         = 2  
\*       INTERNAL\_ERROR       = 3  
\*       OTHERS               = 4  
              .  
    IF sy-subrc <> 0.  
\* Implement suitable error handling here  
    ENDIF.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SMFORM1  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
  
\*# this program is for calling smart form from this program\*\*\*  
\*\*\*this program is known as driver program for smart form  
REPORT ZVAISH\_SMFORM1.  
  
Data : LV\_MATNR type MATNR,  
  
 lv\_fm\_name type RS38L\_FNAM. "default function module name. we are taking in local variable because while delivering and production deployment function module may change  
data : LT\_TAB type ZVAISH\_TT\_MAT.  
  
 SELECT-OPTIONS S\_MATNR for LV\_MATNR.  
 PARAMETERS p\_lifnr TYPE LIFNR.  
  
  
 START-OF-SELECTION.  
 select matnr mbrsh mtart meins brgew ntgew  
   into table  LT\_TAB  
   from mara  
   where matnr in s\_matnr.  
  
 CALL FUNCTION 'SSF\_FUNCTION\_MODULE\_NAME'  
   EXPORTING  
     formname                 = 'ZVAISH\_SMARTFORM1'  
\*    VARIANT                  = ' '  
\*    DIRECT\_CALL              = ' '  
  IMPORTING  
   FM\_NAME                  =  LV\_FM\_NAME  
\*  EXCEPTIONS  
\*    NO\_FORM                  = 1  
\*    NO\_FUNCTION\_MODULE       = 2  
\*    OTHERS                   = 3  
           .  
 IF sy-subrc <> 0.  
\* Implement suitable error handling here  
 ENDIF.  
  
 CALL FUNCTION  LV\_FM\_NAME           "'/1BCDWB/SF00000072'  
   EXPORTING  
\*    ARCHIVE\_INDEX              =  
\*    ARCHIVE\_INDEX\_TAB          =  
\*    ARCHIVE\_PARAMETERS         =  
\*    CONTROL\_PARAMETERS         =  
\*    MAIL\_APPL\_OBJ              =  
\*    MAIL\_RECIPIENT             =  
\*    MAIL\_SENDER                =  
\*    OUTPUT\_OPTIONS             =  
\*    USER\_SETTINGS              = 'X'  
     i\_matnr2                   = S\_MATNR-HIGH  
     i\_matnr1                   = S\_MATNR-LOW  
     i\_lifnr                    = p\_lifnr  
  
     TABLES  
       GT\_TAB                   = LT\_TAB.  
\*  IMPORTING  
\*    DOCUMENT\_OUTPUT\_INFO       =  
\*    JOB\_OUTPUT\_INFO            =  
\*    JOB\_OUTPUT\_OPTIONS         =  
\*  EXCEPTIONS  
\*    FORMATTING\_ERROR           = 1  
\*    INTERNAL\_ERROR             = 2  
\*    SEND\_ERROR                 = 3  
\*    USER\_CANCELED              = 4  
\*    OTHERS                     = 5  
           .  
 IF sy-subrc <> 0.  
\* Implement suitable error handling here  
 ENDIF.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SPELL\_AMT  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_SPELL\_AMT.  
  
  
DATA lv\_spell type spell.  
PARAMETERS p\_amt type i.  
  
call function 'SPELL\_AMOUNT'  
  
exporting  
amount = p\_amt  
language = sy-langu  
  
importing  
  
in\_words = lv\_spell.  
write lv\_spell-word.

\*&---------------------------------------------------------------------\*  
 Report ZVAISH\_STRINGS.  
  
  
\*TRANSLATE STRINGS TO LOWER CASE OR UPPER CASE  
PARAMETERS P\_A(10).  
TRANSLATE P\_A TO LOWER CASE.  
write P\_A.  
  
\* PARAMETERS P\_B(10) OBLIGATORY.  
\* TRANSLATE P\_A TO UPPER CASE.  
\* WRITE P\_B.  
  
\* concatenate  
\*PARAMETERS : P\_A(10),P\_B(10).  
\*DATA: lv\_des(20) type c.  
\*CONCATENATE P\_A P\_B INTO lv\_des.  
\*write lv\_des.  
\*  
\*PARAMETERS: P\_A(10),P\_B(10),P\_C.  
\*DATA: lv\_des(20) type c.  
\*CONCATENATE P\_A P\_B INTO lv\_des SEPARATED BY P\_C.  
\*CONCATENATE P\_A P\_B INTO lv\_des SEPARATED BY SPACE. \* this is executed  
\*write lv\_des.  
  
\*SRRLEN  
  
\*PARAMETERS P\_A(10).  
\*DATA:lv\_des type c.  
\*lv\_des = STRLEN( P\_A ).  
\*write lv\_des.  
  
  
  
\*SPLIT  
\*PARAMETERS P\_A(10).  
\*DATA: lv\_des,lv\_des2.  
\*SPLIT P\_A at ',' INTO lv\_des lv\_des2.  
\*write / :lv\_des,lv\_des2.  
  
  
  
  
  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*New Strings\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*\*\*\*\* translate to uppercase  
  
\*PARAMETERS p\_A(12) type c.  
\*Translate P\_A to UPPER CASE.  
\*write p\_A.  
  
\*\*\*\*\*\*\*translate to lowercase\*\*\*\*\*  
\*PARAMETERS p\_B(12) type c.  
\*TRANSLATE p\_B to LOWER CASE.  
\*write p\_B.  
  
\*\*\*\*\*\*\*\*\*concatenate\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*data lv\_c(20) type c.  
\*PARAMETERS p\_A(10) type c.  
\*PARAMETERS p\_b(10) type c.  
\*CONCATENATE p\_A p\_b into lv\_c .  
\*write lv\_c.  
  
\*\*\*\*\*\*\*\*concate separated by space  
\*  
\*data lv\_c(20) type c.  
\*PARAMETERS p\_a(10) type c.  
\*PARAMETERS p\_b(10) type c.  
\*concatenate p\_a p\_b into lv\_c SEPARATED BY space.  
\*write lv\_c.  
\*translate lv\_c to UPPER CASE.  
  
\*\*\*\*\*\*\*\*\*\*Concatenate by \*  
\*data lv\_c(20) type c.  
\*PARAMETERS p\_a(10) type c.  
\*PARAMETERS p\_b(10) type c.  
\*concatenate p\_a p\_b into lv\_c SEPARATED BY '\*'.  
\*write lv\_c.  
\*translate lv\_c to UPPER CASE.  
  
\*\*\*\*\*\*\*\*\*\*concatenate with user input  
  
\*data lv\_c(20) type c.  
\*PARAMETERS p\_a(10) type c.  
\*PARAMETERS p\_b(10) type c.  
\*PARAMETERS user type c.  
\*concatenate p\_a p\_b into lv\_c SEPARATED BY user.  
\*write lv\_c.  
\*translate lv\_c to UPPER CASE.  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*split\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*PARAMETERS p\_split(10) type c.  
\*data : lv\_a(10) type c,  
\*      lv\_b(10) type c.  
\*split p\_split at '1' into lv\_a lv\_b.  
\*write /: lv\_a ,lv\_b.  
  
  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*offsetting(substring)\*\*\*\*\*\*\*\*\*  
  
\*PARAMETERS p\_a(20).  
\*  
\*write p\_a+0.  
\*data lv\_s(20).  
\*lv\_s(20) = p\_a+2(6).  
\*write lv\_s.  
  
\*\*\*\*\*\*\*\*\*strlen\*\*\*\*\*\*\*\*\*\*  
\*PARAMETERS p\_a(20) type c.  
\*data lv\_a(20) type c.  
\*lv\_a = strlen( p\_a ).  
\*write lv\_a.  
  
  
  
\*\*\*\*\*\*\*\*\*\*\*Replace\*\*\*\*\*  
\*PARAMETERS p\_a(20).  
\*data lv\_a(30).  
\*if p\_a CA 'I'.  
\*  REPLACE 'I' WITH 'O' INTO lv\_a. "into local variable not possibe only in p\_a  
\*    write lv\_a.  
\*  ENDIF.  
\*   write P\_a.  
\*  
\*   PARAMETERS p\_a(20).  
\*data lv\_a(20).  
\*if p\_a CA 'I'.  
\*  REPLACE 'I' WITH 'O' INTO p\_a. "into local variable directly is not possibe only in p\_a  
\*lv\_a = p\_a.  
\*  ENDIF.  
\*   write P\_a.  
\*   write lv\_a.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SUBR  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_SUBR.  
INCLUDE ZVAISH\_INCLUDESTRU.  
  
  
  SELECT-OPTIONS s\_kunnr for lv\_kunnr.  
  PARAMETERS p\_land1 type kna1-land1.  
  
  
PERFORM SUB.  
PERFORM SUB2.  
  
  
\*  
\*&---------------------------------------------------------------------\*  
\*& Form SUB  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM sub .  
    select kunnr name1 land1 ort01  
    into table lt\_tab  
    from kna1  
    where kunnr in s\_kunnr  
    and land1 eq p\_land1.  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*& Form SUB2  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM sub2 .  
    loop at lt\_tab into wa\_tab.  
      write :/ wa\_tab-kunnr,  
      wa\_tab-name1,  
      wa\_tab-land1,  
      wa\_tab-ort01.  
  
      ENDLOOP.  
  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SUBR\_PARA  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_SUBR\_PARA.  
\*Include ZVAISH\_SUBR\_INCLUDE.  
  
DATA A type i value 20.  
  
\*PERFORM SRPV USING A.  
\*write A.  
\*\*&---------------------------------------------------------------------\*  
\*\*& Form SRPV  
\*\*&---------------------------------------------------------------------\*  
\*\*& text  
\*  
\*FORM srpv  USING  VALUE(p\_a).  
\*  p\_a = 50.  
\*  
\*ENDFORM.  
  
  
PERFORM SRPR CHANGING A.  
write A.  
\*&---------------------------------------------------------------------\*  
\*& Form SRPR  
\*&---------------------------------------------------------------------\*  
\*& text  
  
FORM srpr  CHANGING p\_a.  
p\_a = 50.  
ENDFORM.  
  
\*PERFORM SRPVR CHANGING A.  
\*write A.  
\*  
\*FORM srpvr  CHANGING VALUE(p\_a).  
\*  P\_A = 50.  
\*  
\*ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_SUM  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_SUM.  
  
PARAMETERS : lv\_a type i,  
lv\_b type i.  
  
data lv\_sum type i.  
  
CALL FUNCTION 'ZVAISH\_FUNCMODULE'  
 EXPORTING  
  
   I\_A           = LV\_A  
   I\_B           = LV\_B  
 IMPORTING  
   I\_SUM         = LV\_SUM.  
  
   WRITE LV\_SUM.  
          .

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_TRANSPORT\_PROGRAM  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_TRANSPORT\_PROGRAM.  
  
  
write: / 'hii i am transport program'.  
write: 'i need to be transported'.

\*&---------------------------------------------------------------------\*  
\*& Report ZVAISH\_VBAK\_INTERFCLS\_PROGRAM  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT ZVAISH\_VBAK\_INTERFCLS\_PROGRAM.  
  
data : lv\_vbeln type vbeln\_va.  
select-OPTIONS s\_vbeln for lv\_vbeln.  
  
data: lt\_tab type ZVAISH\_TT\_VBELN.  
data obj type REF TO ZVAISH\_VBAK\_INTERFACE\_CLS.  
create object obj.  
CALL METHOD obj->zvaish\_vbak\_interface~vbak\_method  
  
 EXPORTING  
    ivbeln\_low  = s\_vbeln-low  
    ivbeln\_high = s\_vbeln-high  
  IMPORTING  
    etab        = lt\_tab  
    .  
  
  
 cl\_demo\_output=>display( lt\_tab ).